Masters Program in Geospatial Technologies

SHIP RECOGNITION ON THE SEA SURFACE USING AERIAL IMAGES TAKEN BY UAV:

A Deep Learning Approach

Laxmi Thapa

Dissertation submitted in partial fulfilment of the requirements
for the Degree of *Master of Science in Geospatial Technologies*

NOVA IMS Information Management School
Universitat Jaume I
Westfälische Wilhelms-Universität Münster

SHIP RECOGNITION ON THE SEA SURFACE USING AERIAL IMAGES TAKEN BY UAV: A Deep Learning Approach

Dissertation supervised by:

Professor Victor José de Almeida e Sousa Lobo, PhD
Portuguese Naval Academy
Instituto Superior de Estatística e Gestão de Informação,
Universidade Nova de Lisboa
Lisbon, Portugal

Assistant Professor Mauro Castelli, PhD
Instituto Superior de Estatística e Gestão de Informação,
Universidade Nova de Lisboa
Lisbon, Portugal

Joaquín Torres-Sospedra, PhD
Institute of New Imaging Technologies,
Universitat Jaume I Castellón de la Plana, Spain

February 28, 2019

ACKNOWLEDGEMENTS

I would like to extend my sincere gratitude and appreciation to supervisor Prof Dr. Victor Lobo for his persistent support, motivation and supervision. His continuous encouragement and flexible working practices had always created favorable research environment along with his constructive suggestions and guidance always orienting me towards right direction in thesis. He was not only helping on my research but had also been caring whether I am living comfortably in Lisbon. Thank you so much for inspiring me to live an organized life during this six months!

My deep appreciation and gratefulness are also to supervisor Professor Dr. Mauro Castelli for always standing and helping in coping with the challenges I had to faced during thesis period; thank you so much for providing access to server for the high computational performances! Your guidance, critical comments and support throughout the research period have been among the main building blocks for coming out with this thesis work. Likewise, I am thankful to my co-supervisor Professor Joaquin Sospedra for his detailed constructive comments, valuable suggestions, motivation and useful resources for the research.

I am grateful to Professor Marco Painho for his productive comments and suggestions throughout this semester for conducting this thesis. You have always been proactive in solving every single issue troubling not only during the research work but also during the whole study period in Nova IMS. Likewise, I am thankful to Dr. Joel Silva for his valuable time for suggestions and support during the early days of thesis. Similarly, my sincere thanks to Erasmus Mundus Masters course and all three partnering universities for this wonderful opportunity to learn in a completely multi-cultural environment. Additionally, the administrative support received from Erasmus office and Academic services of NOVAIMS and IFGI always helped me in focusing more on the studies. Thank you to all these welcoming people for making my international life easier!

I am equally thankful to all the friends of this masters course for your awesome company. What an amazing family we all have become now! Also, this last semester with the Lisbon Originals-Arman, Roberto, Nicodemus, Duarte, Mitzi, Stefana and William, had really been the most connected one. The whole thesis period had only been easy with your crazy company filled with high motivation, jokes and smiles no matter how bad was the situation. Kudos to you all!

I am also thankful to my friends Shrijana Panta, Sanjeev, Eliza, Sainju, Jyoti, Manisha and everyone who had been continuously cheering me during this international studies and, had been helping directly and indirectly for completing the course. Also, Thank you Sanjeevan dai, Kushal Ang and Manish for the last minute help on proof reading and improving this document.

I have no words to express respects and love to my family; my emotional but courageous mother for her unconditional love, dedication and exceptional patience to make

me and, all of us in the family always connected, my father and sister who always listened my studies and research progress even without understanding a single word, brother in law, Hari for always standing with me, brothers- Santosh for always being accountable to me and whole family (your proof reading has been very helpful), and Amrit for your critical thoughts on my research and always encouraging me to study, nephews-Darpan and Darsish for refreshing me with their smiled filled talks every time. Though I am physically far, your emotional support, motivation and faith on me have always inclined me towards studies. This thesis is dedicated to this amazing family of mine.

SHIP RECOGNITION ON THE SEA SURFACE USING AERIAL IMAGES TAKEN BY UAV: A Deep Learning Approach

ABSTRACT

Oceans are very important for mankind, because they are a very important source of food, they have a very large impact on the global environmental equilibrium, and it is over the oceans that most of the world commerce is done. Thus, maritime surveillance and monitoring, in particular identifying the ships used, is of great importance to oversee activities like fishing, marine transportation, navigation in general, illegal border encroachment, and search and rescue operations. In this thesis, we used images obtained with Unmanned Aerial Vehicles (UAVs) over the Atlantic Ocean to identify what type of ship (if any) is present in a given location. Images generated from UAV cameras suffer from camera motion, scale variability, variability in the sea surface and sun glares. Extracting information from these images is challenging and is mostly done by human operators, but advances in computer vision technology and development of deep learning techniques in recent years have made it possible to do so automatically. We used four of the state-of-art pretrained deep learning network models, namely VGG16, Xception, ResNet and InceptionResNet trained on ImageNet dataset, modified their original structure using transfer learning based fine tuning techniques and then trained them on our dataset to create new models. We managed to achieve very high accuracy (99.6 to 99.9% correct classifications) when classifying the ships that appear on the images of our dataset. With such a high success rate (albeit at the cost of high computing power), we can proceed to implement these algorithms on maritime patrol UAVs, and thus improve Maritime Situational Awareness.

KEYWORDS

Ship Recognition

Classification

UAV Images

Deep Learning

Deep Convolutional Neural Networks

Transfer Learning

Maritime Surveillance

Index Of The Text

Index Of Figures

Index Of Tables

Acronyms

AIS Automatic Identification System.

ANN Artificial Neural Networks.

AODN Accurate Object Detection Network.

CFAR Constant False-Alarm Rate.

CNN Convolutional Neural Network.

COLREGs International Regulations For Preventing Collisions.

CPU Central Processing Unit.

CRFs Conditional Random Fields.

DFPN Dense Feature Pyramid Network.

ELU Exponential Linear Unit.

FlIR Forward Looking Infrared.

GEO Group On Earth Observation.

GPU Graphical Processing Unit.

HMM Hidden Markov Models.

HOG Histogram of Oriented Gradients.

HSF Hierarchical Selective Filtering.

IACS International Association Of Classification Societies.

ILSVRC Imagenet Large Scale Visual Recognition Challenge.

IMO International Maritime Organization.

kNN k-Nearest Neighbors.

LR Learning Rates.

LRIT Long-Range Identification And Tracking.

MaxEnt Maximum Entropy.

MDA Maritime Domain Awareness.

MS-OPN Multi-Scale Object Proposal Network.

MSA Maritime Situational Awareness.

NIR Near Infrared.

OBIA Object-Based-Image-Analysis.

RADAR Radio Detection and Ranging.

RAM Random Access Memory.

RCNN Regional Convolutional Neural Networks.

RCPA Robust Principal Component Analysis.

ReLU Rectified Linear Unit.

RNMS Rotational Non-Maximum Suppression.

ROC Receiver Operating Characteristic.

SAR Synthetic Aperture Radar.

SC-RPCA Segmentation Constrained Robust Principal Component Analysis.

SFM Structure From Motion.

SGD Stochastic Gradient Descent.

SSD Single Shot Multibox Detector.

SVM Support Vector Machine.

TPU Tensor Processing unit.

UAS Unmanned Aerial System.

UAV Unmanned Aerial Vehicle.

VMS Vessel Monitoring System.

VOC Visual Object Classes.

Chapter 1

Introduction

1.1 Contextual Background

The marine ecosystem has always been of interest to navigation agencies, countries and environmental agencies for activities like traffic management and safe navigation, border control, defense and national security, fisheries management, maritime spatial planning, marine pollution, irregular migration and maintaining balanced marine ecosystem (Hartemink, 2012), (Kanjir et al., 2018), (Liu et al., 2017a). Most human activities occur on the sea surface: transportation is done using ships of various types (Zheng et al., 2014); recreational activities will involve jet-skis, rubber boats, canoes, buoys, ships, swimmers or human bodies floating on its surface (Hartemink, 2012); the majority of fishery activities, that are prime source of food around the globe, is done with different types of ships; gas and oil explorations are another potentially valued economic activity that can be observed on the sea surface. Marine activities are continuously increasing with the advancement in marine exploration, transportation, and thus induced economic benefits. However, such growing outreaches are reportedly accompanied with illegal, potentially jeopardizing and ecosystem-unfriendly activities; haphazard fishing activities even in banned areas, oil spills on the sea surface, illicit monitoring over the coastal waters of countries are among such activities. Additionally, the maritime surface has even become the carrier of land-based debris like plastics, metals, glasses that are transported to the marine environment through different acts like urban runoff, sewer overflow, industrial and littering activities (Ediang and Ediang, 2013). This debris is considered among the agents of marine pollution for putting real threats on health, biodiversity, and productivity of marine biota (Buhl-Mortensen and Buhl-Mortensen, 2017). Such coastal engagements characterized by both the beneficial and detrimental activities are driving the attention of the global

community. Various organizations like Group on Earth Observation (GEO), International Maritime Organization (IMO), European Maritime Safety Agency (EMSA), coastal bordered countries and academic institutions are closely monitoring, observing and implementing various systems for the betterment of the maritime environment. Activities to understand maritime environment have been termed as Maritime Domain Awareness (MDA) (Valavanidis and Vlachogianni, 2012), (Hartemink, 2012) or Maritime Situational Awareness (MSA) for more localized picture. MDA and MSA require heterogeneous information (Kanjir et al., 2018) and rely on object monitoring and detection. The most dominant objects over the sea surface used either as the means of carriers for transportation, fishing, surfing, rescue operation, monitoring or supervision are ships or vessels.

Ships are categorized in different types and classes based on the design, constructional structure, and purpose of their usages. These classifications help in ship identification, safety management, and maritime traffic control. Various classification societies are established in different coastal regions and countries that issue classification certificate for the ships with the aim of maintaining maritime safety by setting the technical standards and rules for designing, constructing and maintaining ships. International Association of Classification Societies (IACS) formed as a non-profit membership organization of classification societies is among such actively engaged societies that is technically supporting IMO in its maritime research and development and has even set up compliance rules and standards for vessel classification design and construction along with its twelve of the member societies. Additionally, IMO has published International Regulations for Preventing Collisions at Sea 1972 (COLREGs) that governs ships with common and consistent navigation rules internationally depending upon their classes. Also, the European Code for Navigation and International Sailing Federation is actively working for safe maritime navigation. So, maritime surveillance that basically involves identifying ships on the sea surface, recognizing their classes, monitoring and tracking them visually can be crucial in knowing the activities ships are conducting and intentions of their usage thereby supporting better navigation, controlling illegal activities and detecting oil spills (Gallego et al., 2018).

Environment monitoring and surveillance has been one of the potential applications of remote sensing for the high resolution, qualitative data it provides through spaceborne and airborne sensors system (Li et al., 2018). The Landsat Satellites in 1977 followed by Synthetic Aperture RADAR (SAR) in 1978 through to the first SeaSat Satellite are milestones in earth observation including marine surfaces. Based on the globally followed regulations, different systems like Automatic Identification System (AIS) for short-range operation, Long-Range Identification and Tracking (LRIT), Vessel Monitoring System (VMS) are widely used automatic reporting systems for the ships/vessels. But, not all ships and fishing vessels, especially those with less than 300 tons, are mandated for these systems and there have been cases reported of not using or spoofing the reports to mask illegal activities (Kanjir et al., 2018). Besides,

these systems are designed for a specific purpose like communication transmission or monitoring only the fishing vessels. The weather independent features of SAR providing day-night cloud-free images from RADAR signals have been used widely for ship monitoring and detection. But, it has limitations with low visit time and a smaller number of operational SARs due to the high cost associated with it. Space deployed satellites with higher (commercial) and lower (free) spatial resolutions are available in abundance with wide area coverage over the coastal surface. However, weather dependency and data capturing based on their rotational time and higher computational cost associated with the continuous observations constitute restrictions. The improvements in sensor technology and unmanned aerial vehicles with sophisticated hardware and battery offering longer performing capacity are making them alternative solutions for the surveillance and data capturing tasks requiring low area coverage with higher spatial resolution at an affordable cost. Nowadays, UAVs are widely used in different fields like urban planning, natural disaster assessment, traffic management, surveillance activities for security and safety around the globe, including all the surfaces - airspace, land and marine environment (Li et al., 2018). The flexibility of integrating need-based sensor systems like optical or thermal, on-demand usability and the easy autonomous operation supporting emergency situations like search and rescue are additionally supporting their use in the maritime environment.

The increasing use of aircraft and unmanned vehicle systems for monitoring sea surfaces have produced a huge amount of data and images with detailed information about the earth surface. Monitoring these image data manually and extracting useful information has become difficult using traditional handcrafted methods because of the need for real-time or near real-time performances with high accuracy. Different machine learning and deep learning techniques have been developed to ease automatic feature extraction, object classification, recognition, and detection. Machine learning techniques have involved shallow architectures with one layer for feature transformation and are effective for simple well-defined problems (Deng, 2012). But the real-world applications like ship classification and recognition from optical images having background features like light illuminance and waves require deeper structure capable of learning and extracting features properly with high accuracy. Deep Learning has emerged as a solution in this context allowing multilayer hierarchical architectures for feature learning, classification and pattern recognition. Convolutional Neural Networks (CNNs) are one of the Deep Learning techniques renowned for their outperforming image classification accuracy since the start of this decade, and is continuously advancing with the availability of higher computational power and graphical processing units (GPUs). But, requirements of larger datasets for training in a sophisticated computing environment is still hindering its usability to capacitate common individual researcher at ground level. The concept of transfer learning has been materialized as an alternative to combat these challenges; it involves transferring the learnings obtained by training the chosen network model on large datasets applied

for different usages to another application of similar nature with less dataset.
This thesis focuses on classification and recognition of ships contained in images generated from videos captured by UAVs using Deep Learning with Convolutional Neural Network and transfer learning. The video datasets obtained from Seagull Project taken over different areas of the Atlantic Ocean are the primary sources of image sequences.

1.2 Problem Statement and Motivation

Most of the research on marine ship monitoring involves extraction and classification of image features using different shallow structure algorithms like Histogram of Oriented Gradients, exemplar Support Vector Machine (Chua et al., 2014) eigenvalue analysis with principal component analysis (Pietkiewicz and Matuszewski, 2018). These techniques are lacking the demanding accuracy and, even if achieved are at the cost of performances and more workloads in terms of parameters. But, advancement in computer vision technology and development of deep learning techniques, particularly convolution neural networks (CNNs) has been offering remarkable performances in the field of image recognition with strong feature learning ability, fewer model training parameters and high recognition accuracy (Kumar and Sherly, 2017), (Wang et al., 2018). Among some researches done in a maritime environment using CNNs, most are done either with SAR Data or satellite-based optical images (Tang et al., 2015) with more focus on detection than the recognition. In case of classification also, these are performing binary classification like Gallego et al., 2018 to identify if ship is present or not. Very few researches are done using CNNs with UAS captured data; Moreover, data used are normally taken in similar environmental conditions with the same sensors. Even at these scenarios, the primary challenges for the videos and images taken with UAS over the sea surface are characterized with scale variability, movements of UAVs, wave crests and sun glare difficult to identify ships based on their size, shape or textures (Ribeiro et al., 2017). So, the novelty of this thesis lies on exploring the potentialities of deep learning that have been put forward with deep CNNs in image classification and recognition scenarios to recognize the class of ship from images generated by the videos captured by different visible and infrared sensors deployed with UAS over different areas of Atlantic Ocean in different time period.

1.3 Research Aim and Objectives

The aim of this thesis is to perform multi-classification of ships and their recognition by training deep convolutional neural networks on the aerial images captured by UAS over the sea surface using the concepts of transfer learning. To achieve this main aim, specific objectives have been set as follows:

- Review existing state of art on the sea objects classification, recognition, deep learning, CNNs and transfer learning.
- Choose suitable Deep CNN architecture and pre-trained weights, design and implement algorithms using transfer learning techniques for multi-classification of ships and their recognition.
- Performance evaluation and review of the proposed approach with the existing approaches.

1.4 General Methodology

The schematic diagram 1.1 below provides an overview of the overall methodology applied in this thesis. It basically involves reviewing existing works and determining suitable deep learning methods for image recognition along with its parameters and then using the identified methods with suitable modification to recognize ships present in images with their respective classes. Lastly the model obtained from modification will be evaluated using different approaches like precision-recall rate and confusion matrix. Detail methodological description about the methods adopted can be found at Chapter 4.

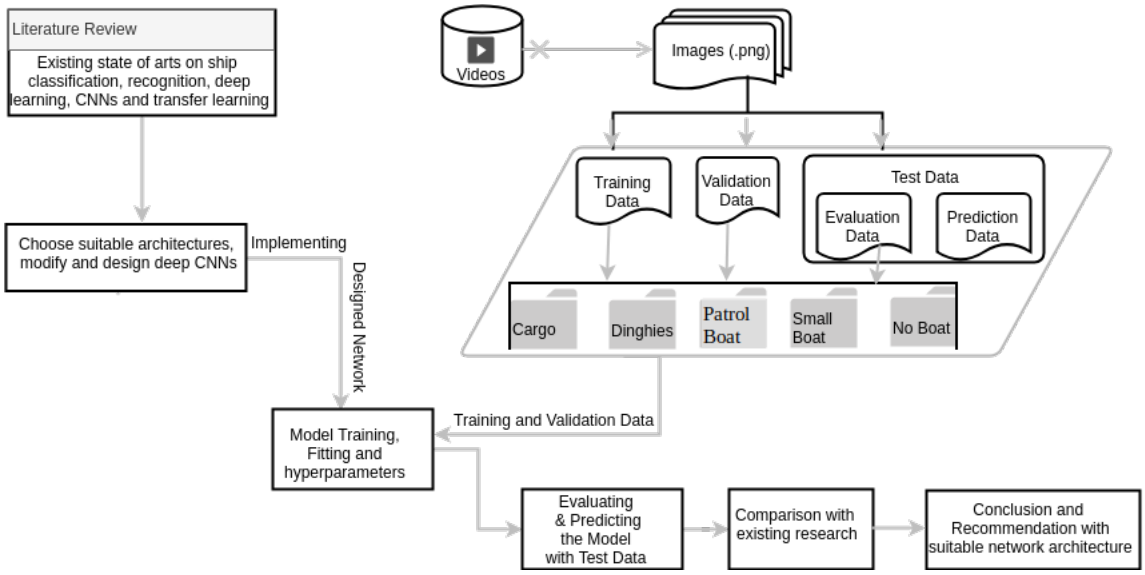


Figure 1.1: Methodological Overview

1.5 Contribution

This thesis will use deep learning to address the current gaps and challenges mentioned in section 1.2 for recognizing ships on the sea surface. The main contribution of this thesis consists of:

- Exploring the possibilities of using deep learning for recognizing ships on images generated from a non-static UAV mounted with camera having Visible and Infrared sensors.

- Proving that the final model trained on the thesis dataset can be used as a product to make predictions or to recognize ships present in images.

1.6 Thesis Organization

This thesis consists of seven chapters describing the entire activities carried out to complete it. Chapter 1 entails foundation of thesis starting with the first section briefing about contextual background on the need for ship monitoring and recognition followed by the existing platforms and techniques used for this purpose. It further highlights the need for classifying ships, existing methods, and techniques used for classification and gives a short description of this thesis work; second subsection mentions about the underlying gaps in existing research on the ship classification and recognition followed by the challenges this thesis has taken using the Seagull data; it also mentions briefly about the proposed method of CNNs; the third subsection states the main aim of this thesis along with the specific objectives set to achieve it; subsection four gives an overview of the proposed methodology defined to achieve the research aim and objectives formed through contextual background and problem statement.
Chapter 2 consists of a literature review explained with the related work on the thesis in the first part and afterward contains theoretical definition and description of the terms, terminology, and methods used throughout the thesis.
Chapter 3 explains the detail information about dataset used in this thesis by describing the source of data and platform used for data collection. It further describes how data has been compiled and prepared to suit the methodology requirement of this thesis. It also describes resources used in this thesis including both the software and hardware.
Chapter 4 puts forward the idea of the methodology proposed based on the literature review with related work having the start of art performances to achieve the aims and objectives set in Chapter 1. Its latter section describes in detail the chosen architecture and its modification with suitable parameters and hyperparameters. It also describes the designing of the algorithm developed for the thesis.
Chapter 5 showcases results and briefly mentions the results. It also evaluates the accuracy obtained along with network performances, computational power and time required for the model´s execution.
Chapter 6 discusses the significance of results. It further presents the comparison of this thesis with the similar works done before and presents the differences between these methods and distinction of proposed method with them.

Chapter 7 explains the conclusion obtained from this thesis together with the limitations and recommendations for future works.
The content afterward consists of Annexes I to supplement the contents described shortly in the main chapters for page limitation.

CHAPTER 2

LITERATURE REVIEW

This section consists of comprehensive review on existing state of art on the object recognition techniques for sea surface based on the existing research and projects. The first part starts with background on the importance of sea surface monitoring and, objects detection followed by the platforms and tools used for these tasks. The second part discusses the conventional approaches used for sea surface object detection starting with the satellite images and then UAS images The third part presents about deep learning techniques, particularly Convolutional Neural Networks and different techniques used on CNNs for the better performances and accuracy used for the detection. The fourth and last part explains the theoretical definition and insights on deep learning, CNNs, and other the technical terms used in the research.

2.1 Related Works

2.1.1 Sensors platform used for ship recognition

Different sensors have different features that determine their applications; spatial resolution, update rate, range, coverage, persistence, latency, and cost are among the major concern. The most widely used sensors in marine surveillance are optical, infrared and radars deployed either on satellites, aircraft/UAS, ships or shores (Kanjir et al., 2018). Radar is the typical technology for monitoring and detecting ships that is in use since the 1990s; Synthetic Aperture Radar (SAR) is one of the most popular and widely used maritime monitoring and detecting Radar techniques for its ability to capture images independent of weather and daylights. It is even independent of the distance to the observed object. However, SAR has the limitations of being highly prone to intrinsic noise, low spatiotemporal coverage with limited number of satellites, long revisit cycle because of a smaller number of SAR satellites, difficulty to detect

small objects (Liu et al., 2017a) and, recognition of false alarms leading to difficulties in classification (Kanjir et al., 2018). Satellite-based sensors are popularly used for the wide area accessibility, remote access, systematic monitoring of data continuously, and availability of larger data collection (Kanjir et al., 2018) for time series analysis. Landsat, SPOT, QuickBird, IKONOS, Google Earth are among the highly used satellite sensors for maritime object recognition (Kanjir et al., 2018). The free availability of Sentinel-1 and Sentinel-2 optical satellites being operated under European Space Agency Earth Observation Missions[1] has further increased the potential market of optical satellites.
For the usability on demand basis like emergency search and rescue, advancements in microelectronics (Bejiga et al., 2017), easily transportable, economically affordable and improvements in sensors technology and battery system for the integration with UAS system to achieve desired data resolutions for longer duration, applications of UAV-based sensors have increased extensively in the recent decades. Development of autonomous operating system offering many automatic facilities like flight take-off and landing, aerial refueling and route planning with higher levels of accuracy (Al-Kaff et al., 2018) have induced the concerned service providers and users to apply UAVs as the mundane means for navigation and surveillance activities. Also, sea surface monitoring and vessel detection from UAVs are increasing rapidly and so are the research on these fields growing (Dolgopolov et al, 2017, (Xu et al., 2014). Johnston, 2019 has discussed that UAVs have been used significantly in studying marine wildlife particularly for large marine creatures like whales, sea turtles, sharks.

2.1.2 Images used for sea object recognition

Hyperspectral sensors are the emerging remote sensing technology that uses imaging spectrometer to extract spectral information of a spatial area but possesses low spatial resolution when observed from orbit and involves complex processing. So, sea surface object detection, even the ship is difficult to detect from these images though there is less research done for its application in maritime (Kanjir et al., 2018), (Wang et al., 2016). Thermal infrared sensors used normally for night time image capture obtain emission from captured objects themselves unlike depending on solar illumination (Kanjir et al., 2018) whereas if used during the day, objects are detected based on the temperature differences between sea and surface objects (Wang et al., 2016). These sensors have the limitation of low resolution when measured from satellites and, suffers from atmospheric clouds and moisture when measured from the atmosphere. Optical images observed in the visible spectrum are easily detectable by the human eye; photos captured from a normal camera are also optical images. These images are consistent and enriched with essential information for feature extraction (Kadyrov et al., 2013) (Kanjir et al., 2018) including vessel detection and classification for economical price

[1] https://earth.esa.int/web/guest/missions/esa-eo-missions

and simplicity in structure (Lan and Wan, 2009). Further, these images offer high spatial and spectral resolution useful for detecting smaller objects (Pegler et al., 2007) though are greatly affected by weather and sun reflection on the water (Kanjir et al., 2018). There have been increasing trends in mounting video cameras and different sensors with aircraft and UAS, even on buoys or on floating platforms as these offer easy installation and maintenance (Kanjir et al., 2018); object detection in a such system involves the analysis of the images taken separately or the image frames generated from the videos and, are the data of interest for this research.

2.1.3 Sea Surface Object Recognition Approach

The object recognition process on sea surface involves mainly detecting the objects in the images, often referred as feature extraction and distinguishing the extracted features by discriminating them with water on the sea as non-water objects and assigning the class of the object, referred as object classification (Nie et al., 2017). The contents below initially present the traditional object recognition methods followed by modern approach of deep learning for classification and recognition.
The most widely used and researched sea surface objects detection methods use SAR images that involve the use of algorithms like constant false-alarm rate (CFAR) detector with the combination of Gauss distribution, k-distribution, and Gamma distribution or their individual uses (Liu et al., 2017a) for the feature extraction. Object detection from optical images in the past involved traditional handcrafts methods involving manual feature extraction from images based on shapes, textures and physical properties. The first research according to Kanjir et al., 2018 on ship detection was done by McDonnell and Lewis, 1978 using the Landsat imagery; McDonnell and Lewis, 1978 inspected Landsat CTT printouts and put forward a threshold-based approach of detecting the ships by using total pixel numbers occupied by the ships in the MSS band, orientation of these pixels, their maximum and total pixel radiances values. Since then, many research have been carried out for ship detection using different methods; Kanjir et al., 2018 has reviewed 117 papers for optical satellite images, some of the methods are mentioned below: 1993 used transform domain method with high/low pass filter on SPOT XS and Landsat TM images; ship detection based on shape and texture were used with different considerations like local image statistics with spatiotemporal features (Pegler et al., 2007), shape constraints (Wang et al., 2016), spatio-spectral template enhanced with weighted Euclidean distance metric (Pegler et al., 2007), cumulative projection curve with Mahalanobis distance (Hu and Wu, 2008), region based, shape-prior segmentation, mathematical morphology (Zhu et al., 2010), local binary patterns (Ji-yang et al., 2016), object-based image analysis (Li et al., 2016); this method though is enriched with spectral information with good accuracy for object detection suffers the issues of false alarm candidates. The threshold based method is used in different ways like histogram-based segmentation, canny edge and Fourier transform (Hu

and Wu, 2008), (Li-xiaa et al., 2010) hierarchical clustering merging algorithm (Hong et al., 2007), adaptive threshold segmentation ((Hu and Wu, 2008) component tree image algorithm (Xu et al., 2011), (Zuo and Kuang, 2011), (Guo and Zhu, 2012) these are suited mostly for the smooth sea surface. Salient based methods like multiscale enhancement method (Li et al., 2016), hypercomplex frequency domain and phase quaternion fourier transform (Li et al., 2016), histogram-based contrast method (Liu et al., 2017b) are good for heterogeneous sea surfaces but may not be the good option in presence of high clutter on the image as it can lead to false alarm. Other methods like statistical though is quick result generative, it needs a good knowledge of the tools being used; transform domain method is not the good choice for high heterogeneity; anomaly detection methods are good for threshold and sea surface heterogeneity cases but shows bad performances for the near coastal ships.

Other sea surface objects have also been detected and an airborne system like aircraft or UAVs are normally integrated with electro-optical sensors having visible, infrared or hyperspectral spectrum cameras to capture the videos or images. The similar approaches like that of optical satellite imagery and or with some developments and customization have been used for recognition purposes. Also, earlier days of UAS object detection were facilitated with computer vision techniques to some extent. Borghgraef et al., 2010 discusses problems associated with background subtraction algorithm and showed out performances by the algorithms like behavior subtraction and ViBe for detecting floating objects, particularly free-floating mines on the sea surface. Zheng et al., 2014 applied saliency detection method with Locally Adaptive Regression Kernels using self-resemblance techniques. Shin et al., 2016 presents the objects detection method that involves a coarse-to-fine resolution approach by customizing the stereo-vision based techniques and top-view grid method. Ventura et al., 2018 used open software-based structure-from-motion (sfm) techniques for orthomosaics followed by multi-resolution segmentation algorithm and spectral difference segmentation algorithm for Object-based-image-analysis (OBIA) in the commercial software for coastal mapping and classification. Seymour et al., 2017 used threshold method with high pass filter in ArcGIS model builder programming environment to detect seals in thermal imagery from fixed-wing UAS; this model used preprocessed rectified images and is suitable only for the images having single species. Leira et al., 2015 applied simple edge detector techniques for detecting the marine objects from the thermal imaging camera on a low-cost fixed-winged UAV and nearest neighbor classifier was applied for classification considering object size, temperature, and overall structure. These methods require a higher extent of human inputs and good computational skills and platforms.

Developments in computer vision technology and machine learning algorithms have increased the level of automaticity in object detection and classification with high performances; methods like co-training model (Guo et al., 2015), random forest method (Huang et al., 2015), sparse representation and Hough voting (Yokoya and Iwasaki,

2015), rotation and scale invariant method (Lin et al., 2017), Line segment detector Yao et al., 2016 have been used previously for these purposes (Kanjir et al., 2018). Prasad et al., 2016 presents the challenges like presence of occlusion, orientation, scale, variety of objects and their motion patterns along with the variations in weather and illumination condition associated with the maritime image processing from videos generated from cameras; they also discussed briefly the background object detection using Gaussian mixture, Gaussian background model and self-balancing sensitivity patterns analysis. Yu et al., 2015 proposed the context-driven Bayesian saliency model for detecting small and dim objects on FLIR images having sea clutter. These methods usually involve preprocessing tasks with the object features definition. Besides, these machine learning techniques offer the non-linear feature transformations, only for a single layer.

However, these methods including, hidden Markov models (HMMs), conditional random fields (CRFs), maximum entropy (MaxEnt) models, support vector machines (SVMs), logistic regression and kernel regression, usually referred as shallow structure do not support multi-layers features. Also, these traditional machine learning techniques require explicit object feature definition. But advancements in the deep learning fields, computer vision, and computational platforms have drastically changed the conventional approach of image processing and analysis techniques. The winning image classification approach of Krizhevsky et al., 2012 in ImageNet Large Scale Visual Recognition Challenge (ILSVRC) 2012 popularized the potentialities of deep CNNs for image recognition, detection and particularly for classification (Rawat and Wang, 2017). Since then, lots of research has been carried out on CNNs with improvements in various factors like network architectures, activation functions, optimization techniques, supervision components and regularization mechanisms (Rawat and Wang, 2017).

However, very few of the research is focused on using CNNs for sea surface objects detection; Tang et al., 2015 used compressed-domain framework, deep neural network and extreme learning machine for detecting and classifying vessels from optical SPOT-5 images; Zhang et al., 2015 proposed and discussed the usefulness of S-CNN method developed by combining CNNs with saliency detection method, for ship proposals detection with high recall and good accuracy compared to R-CNN method. Borji et al., 2014 surveyed and discussed the existing methods and state-of-art performance using Saliency object detection methods including classical models like localization, region-based, and Segmentation models along with CNN based deep learning-based models; CNNs based technique is not yet explored much with the multiple images and multi-objects, and implementation for this method also lacks publicly available large datasets.

Ren et al., 2017 introduced novel Region Proposal Network by merging it with Fast R-CNN such that the former network component predicts the object bounds thereby showing the proposed regions and the latter uses the proposed regions for detection. It

is a nearly cost-free approach suitable for near real-time and improves the region proposal quality. The method has been the fundamental base for winning entries of object detection competition like COCO 2015, ILSVRC 2015. Nie et al., 2017 used transfer learned Single Shot MultiBox Detector (SSD) on labeled satellite images through the VGG model to detect ships; the paper shows the higher accuracy obtained by this method than Faster R-CNN with less computational work involved in addressing the multi-scale problems by using features map from multi-layers whereas, the Faster R-CNN uses feature map only from the top layer causing the difficulties in small ship detection by pooling and sampling.

Liu et al., 2017b used CNN method as the ship classifier on Google Earth Images and obtained higher accuracy as compared to SVM and Neural Network. Turner et al., 2016 using optical images for object detection and classification in navy application introduced detection and classification technique called Spatially Related Detection with Convolution Neural Networks to address the spatial configurations of inter-object within images for effective region proposal technique to use with the existing CNNs approach. It highlights the importance of spatial relations to improve accuracy; the results though showed improved classification accuracy has not been able for the remarkable improvements in object detection.

Faster R-CNN is not considered as an efficient method for the densely packed objects detection in practical remote sensing activities (Deng et al., 2018). So, Deng et al., 2018 proposed a new method to overcome this problem based on Residual Networks, known as ResNets; it consists of two subnetworks within it for detecting the object proposal and then its detection by adopting existing Faster R-CNN. With a further continuation, Deng et al., 2018 has also proposed another CNN method that consists of feature extractions using Concatenated ReLU and Inception Module followed by object detection using two sub-networks, multi-scale object proposal network (MS-OPN) and accurate object detection network (AODN) for handling multi-scales and multi-objects respectively. Li et al., 2018 proposed state of art performance method as the regional proposal network based deep CNN to detect inshore and offshore ships on multi-scales using hierarchical selective filtering (HSF); this method is the modification of faster R-CNN architecture that includes CNN for feature extraction, HSF layer to deal with multi-scale deep features for ship detection region and ship detection respectively, and has been named as HSF-Net.

Khellal et al., 2018 proposed a new approach, claimed as state-of-art performance on CNN features learning and classification by introducing Extreme Learning Machine (ELM) for ship detection from infrared images; ELM is discussed as the efficient approach to cope with the problems of back-propagation regarding slower speed and requirement of many hyperparameters. This method considers fully connected layers as a convolutional layer such that only the Convolutional layers are trained with no need for any parameter for Pooling layers. Wang et al., 2018 used the combination of CFAR and CNN methods on processing SAR images for automatic detection of

ships and achieved higher accuracy and computational speed than the multi-threaded and multilevel CFAR algorithm. Yang et al., 2018 proposed multiscale rotational region CNN consisting of Dense Feature Pyramid Network (DFPN), adaptive region of interest Align, rotational bounding box regression, prow direction prediction and rotational non-maximum suppression (R-NMS) to solve the issues of redundant ship detection region, difficulties of dense ships and complexity of the application scenarios. This end-to-end rotational-region-based detection method is also able for predicting berthing and sailing direction of the ship. However, the method suffers from problems of higher false alarms rate resulting in lower precision than the FNN and Faster RCNN methods. Gallego et al., 2018 presented CNN based architecture combined with k-Nearest Neighbour method for the ship classification on MASATI dataset resulting in the state-of-art ship classification method. This method was applied on various network models like VGG-16/19, ResNet, Inception V3 and Xception to detect and classify different kinds of ship like cargo, oil, boat, cruiser; highest classification accuracy was obtained with Xception model that even outperforms existing models and methods. However, this method does not address the issues of multi-sensors.

2.1.4 Improvements in Deep learning

The volume of data is increasing every day and computer vision is broadening its applications with the easy availability of graphics for high computational efficiency in cheaper price; even some of the cloud platform like Google Colab and Floydhub are offering free use with some limitations, for researchers. The researchers are continuously working on the improvements of existing algorithms along with the increasing number of data sets and computational resources for higher performance. The machine learning algorithms used earlier have now been outperformed with the deep learning techniques. Chua et al., 2014 compared three of the classical machine learning algorithms, Histogram of Oriented Gradient, Exemplar-SVM and Latent-SVM with deformable Part Models considering their high performance achieved in Pascal VOC Challenge; HOG is used to derive feature sets as a feature descriptor; exemplar SVM focuses on specificity, not on generality and can involve high computational costs for multiple models training, whereas latent SVM is the extension of the HOG model. The last algorithm performed well among the others in Annapolis Harbor corpus datasets used in the study. Yu et al., 2015 proposed a context-driven Bayesian saliency model that takes into account the contextual information associated with locations and scales of the objects and sea surface in FLIR images. This method is useful to cope up with the scale variance and complicated background existing in the images because of sea clutter and clouds. Leira et al., 2015 discussed on the UAV with machine vision system equipped with thermal imaging camera for real-time object detection, classification, and tracking of objects in the ocean surface. The applied automation involves edge detector and nearest neighbor classifier for objects detection and classification. Li et al.,

2017 proposed Segmentation Constrained Robust Principal Component Analysis (SC-RPCA) for detecting the moving objects having bad weather and changing background in the videos with better performances; this method uses Gaussian Max-Pooling in order to differentiate the foreground objects from dynamic background scenes by estimating the stable-value for each pixel and Segmented Constraints RCPA ensures temporal and spatial continuity into the images.
Audebert et al., 2017 uses the approach of segmenting object first using FCNN, then detecting the object through regression on the bounding boxes and finally classifying them using CNN; they applied this technique for individual vehicles classification. Griffiths and Boehm, 2018 used the concepts of deep learning, especially CNNs for the applications in applied engineering purposes using UAS aerial images; they used three CNNs, two Faster RCNN models based on Resnet and Inception-Resnet and the third is Focal Loss network architecture based on Retinanet, outperforming former two of the network models for detecting the railway track. It does single-class object detection only.

2.1.5 Network Architecture and their choices

Many network models have been developed from different convolutional neural network architecture. Mostly they follow similar design principles with the input as convolutional layer followed by the layer for spatial dimensions downsampling along with the increment in the number of feature maps. The early days´ network architectures like LeNet-5 (LeCun et al., 1998), AlexNet (Krizhevsky et al., 2012) and VGG16 (Simonyan and Zisserman, 2014), referred as classical consists of stacked convolutional layers whereas the modern architecture afterward possesses modifications offering high learnings. The network models like AlexNet, GoogleNet, and Resnet are the state-of-art performing models with a significant breakthrough for image classification (Gallego et al., 2018). These architectures for their high performance have been often adapted as the basis of either feature extractors or creating new network architecture for CNN induced research and computer vision tasks like image classification, object detection, and image segmentation. Each network model has its own features and characteristics. The LeNET-5 model introduced by Yann Lecun in 1998 as the handwritten digit identification system for zip code recognition in postal service is considered as the pioneering of famous CNNs used globally these days; it consisted of 60,000 parameters for training the features. However, the model suffered from fewer applications at that moment because of higher computational performances it required. But CNNs were again rebirthed by Krizhevsky et al., 2012´s AlexNet in 2012 after winning above mentioned ImageNet competition; this model is deeper than LeNet-5 with 60 million parameters generated by 5 convolutional layers followed by 3 fully connected layers; it further uses ReLu instead of Tanh and sigmoid functions used in traditional neural network and introduced Dropout layers to reduce the problem of

overfitting. Subsequent development has been occurring since then with the introduction of newer and modified network architecture favored by the every day advancing computer vision hardware resources. VGG 16 introduced in 2014 by VGG group from Oxford is the improved version of AlexNet with more deeper network architecture; it replaced 11*11 and 5*5 kernel size filters with 3*3 kernel size thereby increasing depth of the network and learning complex features. It uses Dropout and Max-pooling techniques and ReLU activation functions. However, deeper networks were costly in terms of higher computational power and time. So, Inception (GoogleNet) introduced by Google researcher team in the same year came out with more modifications and even won the 2014 ImageNet competition; it introduced inception module, reduced the number of convolutional filters, offers use of the different size of convolutions and bottleneck layer by reducing the computational power requirements. Inception even uses global average pooling instead of the last fully connected layers to reduce the total number of parameters but achieving higher accuracy in a short time with more wider network architecture. ResNet came with another idea of having deep residual networks by introducing residual blocks to learn feature maps adjustment more deeply; it solved earlier networks issue with deeper networks causing accuracy saturation and rapid degradation. This network architecture offered better accuracy and performances than the previous architecture and even won the ILSVRC 2015 classification competition. Xception is the refinement of Inception model with 36 convolutional layers offering depthwise separable convolution operation and it achieves higher accuracy than Inception while using the same number of parameters.
More recently, DenseNet has been introduced with the idea of referencing feature maps from the earlier stage of the network to all the subsequent layers resulting in a higher performance with less complexity than ResNet; it reduces the number of parameters and reuses features. NasNetLarge is another network architecture offering better performances. Also, these existing architectures are being revised and have different versions like VGG 19, Inception V3, ResNet101, ResNet152 and many network architectures like FractalNet, SqeezeNet, MobileNet have been developed rapidly. Due to the availability/development of free datasets and higher computational power with GPUs, there might have already been more latest network architecture available publicly with better accuracy than described here. *Applications - Keras Documentation* presents the list of pretrained models on ImageNet dataset available with Keras API according to their accuracy. Computational accessibility and amount of training dataset available play important roles for building a deep convolutional network and choosing network architecture. Among the above-mentioned architecture, this research is using both the classical and modern architecture to observe accuracy obtained on the research dataset: it will use VGG16, ResNet, InceptionResNet, and Xception.

2.1.6 Transfer Learning

CNNs for their excellence in performance for image recognition and classification have been widely used for a large number of datasets but suffers from the problem of overfitting if it is used on the small datasets. Also, it requires high computational power and large memory. It is merely possible for every researcher to find millions of datasets and train it with the advance resources integrated with GPUs and a large number of parameters demanding a long time. In such context, learnings from the pre-trained network on large datasets are considered very useful and are termed as transfer learning. Unlike the traditional machine learning concept of having training and test data from the same feature space and the data distribution (Pan and Yang, 2010), transfer learning accepts the variations and, learnings are shared among different environment for improving the generalization (Goodfellow et al., 2016), accuracy and performance. Transfer learning is based on the concept of applying previously learned knowledge to the different task of similar nature by using the original pre-trained network to update weights on the new training dataset and to extract the features (Ali and Angelov, 2018). It is a time-saving, computationally cost-effective computer vision approach that uses already existed pre-trained models trained with larger benchmark dataset to solve the similar types of problems but on different dataset and scenarios. Applications of transfer learning have been increasing in recent years. Figure 2.1 shows its growing use to elevate machine learning commercially. Razavian et al., 2014 suggested the use of CNN extracted features as the primary input for visual recognition tasks based on the outperforming accuracy he achieved from overfeat extracted features classified by using linear SVM. Kim, 2014 achieved higher accuracy with state-of-the-art improvement on sentence classification using pretrained vectors by extracting the features and fine-tuning few hyperparameters. Shin et al., 2016 discussed the usefulness of transfer learning and achievement of state of art performance on mediastinal LN detection by using the ImageNet pre-trained models. Ali and Angelov, 2018 used the pre-trained CNNs based on the AlexNet Structure to extract the features and then SVM to classify the human faces for anomalous behavior detection.Pan and Yang, 2010 entails theoretical insights on transfer learning, its strategies and applications on the issues like classification and clustering through a survey. Many organizations and researchers are putting efforts on collecting and providing different datasets of images freely as an initiative to encourage academicians, researchers, and data science communities for promoting research on automated tasks that involve the machine and deep learning to accelerate the ongoing developments in artificial intelligence. ImageNet, CIFAR, MNIST, COCO, PASCAL VOC 2012 are among the popular datasets that are trained with a large number of parameters identified through intense research on more than thousands of different object classes with different network models like VGG16, Xception, ResNet. Even many competitions are happening for object classifications, localization and detection on these datasets every year resulting in the new or modification

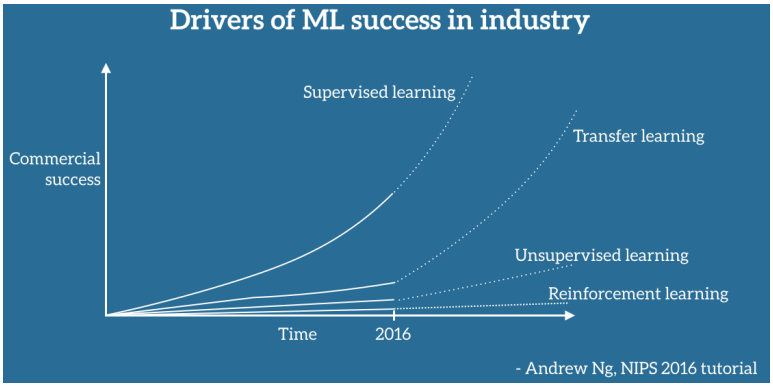


Figure 2.1: Drivers of Machine Learning success in industry[2]

of existing CNN networks with higher accuracy. The results from such competitions like ILSVRC are also available free for further research and improvements. Many researchers are using these network architecture, datasets, and pre-trained models for their research. ImageNet[3] is one of the most popular image database portals maintained by Stanford Vision Lab, Stanford University and Princeton University. It offers free use of 14,197,122 images with human annotation available at the moment and arranged according to WordNet hierarchy with the aim of providing well managed easily searchable images to the researchers around the globe. The ImageNet Large Scale Visual Recognition Challenge[4] (ILSVRC) is the object detection and image classification competition happening every year since 2010 using the 1.2 million images with 1000 categories of objects from ImageNet as the training data. Most of the CNN network architecture mentioned previously with state of art performance are winners of this challenge. Besides, corresponding models from these network architectures trained on this ImageNet dataset and that achieved the highest possible accuracy during the time of challenge have been made available for free use. These weights and thus obtained trained models known as pre-trained models have been offered by different deep learning frameworks like tensorflow, keras, caffe, pytorch as the imagenet weights and, are serving as the key basis of growing applications of transfer learning these days (Kornblith et al., 2018). ImageNet dataset has received global acceptance by the deep learning-related research community for image recognition tasks.

2.2 Theoretical Frameworks and Terminologies

This section briefly mentions the theoritical concept and definition of terminologies used in the training process of a CNN model.

1. **Artificial Intelligence, Machine Learning, and Deep Learning**
   Artificial Intelligence is considered as the human brain influenced intelligence

[3]http://www.image-net.org/
[4]http://image-net.org/challenges/LSVRC/2016/index

learning mechanism that uses a computerized system based upon statistical and computational techniques for extracting and learning the characteristics of an object or system such that it can utilize these learnings or representations for solving similar cognitive problems intelligibly like humans. John McCarthy, one of the pioneers of AI has defined it as *the science and engineering of making intelligent machines that have ability to achieve goals like human do*. The recent research and inventions on automation like self-driving cars, internet search engines, and speech recognition are rooted on the concept of AI.

Machine Learning is considered as the subset of Artificial Intelligence that is powering AI system through data analysis by developing statistical models and algorithm capable of analyzing data, identifying the existing patterns and making decisions without repeatable human interventions and programs. With the every day generated and increased big data in health, finances, marketing, satellite images or/in cloud platforms, hand-crafted data manipulation has been challenging; machine learning has evolved with high computational abilities to support these challenges with automation by using various methods like supervised, unsupervised, active and reinforcement learning to learn the data and make predictions.

Deep Learning is considered as the subset of Machine Learning that has the capability of learning data with more complexity by going more deeper as its name suggests and is highly influenced by the animal nervous system. It consists of multiple layers, usually referred as neural networks that are trained on datasets to learn their features so that they can result in the output with higher accuracy; training more data is considered as the main factor for increasing accuracy of predictions.

2. **Artificial Neural Networks, Convolution Neural Networks, and their functioning**

Artificial Neural Networks (ANN) are biological nervous system inspired neural frameworks consisting of a large number of units called neurons interconnected to each other for processing given inputs, learning their properties and making decisions or predictions based on the learnings during the processing stage. A simple ANN consists of an input layer, hidden layer, and output layer. Every layer consists of neurons and all these neurons are fully connected to each of the corresponding layers.

Convolutional Neural networks are hierarchical neural network system that consists of neurons resembling human visual cortex to make connections between multiple layers, usually referred as convolutional and sampling layers existing in the network. Basically, these are feed forward deep learning neural network algorithm trained through back-propagation techniques and comprise three main layers: input layer consisting of input data with defined size; it is

followed by hidden layers consisting of many convolutional layers, activation functions, pooling layers, fully connected layers, and final output layer (fully connected) displaying the product of input layers processed and trained with neural networks. CNNs are characterized by following features:

- There is spatial local connectivity between neurons of adjacent layers.
- They share weights among the architecture.
- They are shift/space invariant artificial neural network.

Convolutional layer: It is the core building block of a CNN that consists of filters (often called kernels) performing convolution operation across the height and width of the input feature in the initial network layers; mathematically, matrix multiplication is carried out between the no. of filters and the input feature size resulting in 2-dimensional feature map (or activation map). The number of convolutional layers can vary from one to many based on the number of datasets, feature complexities and computational capacity available. Present as hidden layers, they are mainly responsible for extracting features such as edges, colors, orientation of the input data and reducing the image size to ease the learning process with no loss in data properties. The size of a filter sliding over the input data is called stride. Pooling Layer: This layer is introduced to reduce spatial dimensionality of the output from convolutional layer thereby favoring less computational requirements and extracting highly dominant features of the input data. This layer also helps to reduce overfitting. Based on the way of analysis, there are two types of pooling layers: Average pooling: It averages the values of pixels contained in images covered by each of the filters. It reduces noises through dimensionality reduction. Max pooling: It results in the maximum value of pixels contained in the images and covered by the filters. It suppresses noise activation through positional invariance and dimensionality reduction. It is considered a better option than the average pooling.
Output layer: This layer is the fully Connected Layer introduced at the end of a CNN to learn non-linearity in features after the input images are learned from convolutional layers. This layer as the name suggested is fully connected to every neuron or activation maps in the previous layer. Flattening the images extracted as 3-dimensional data into vector takes place before the output layer generates the prediction. Based on the number of iterations applied, this flattening layer undergoes feed forward and back propagation process to reduce the errors and make predictions with high accuracy. This layer is also called the classification or prediction layer. Convolution and pooling layers together serve as the feature extractors, and the last fully connected layer functions as a classifier in a CNN.

3. **Activation and Loss Functions:**

Activation functions are introduced in convolutional layers to produce non-linear outputs without affecting receptive fields of the convolution layer. These are also considered as the decision function of the neuron´s output. There are different types of activation functions like ReLU, eLU, hyperbolic tangent, sigmoid function; the most widely used is the ReLU for its performance efficiency.
Loss function is introduced in the final fully connected layer to determine the discrepancy between the training and predicted output with true labels. Different functions like Softmax, Sigmoid, Euclidean are used as loss function depending on the input-output nature feature dataset.

4. **Parameters and Hyperparameters:**
Parameters though often used interchangeably with hyperparameters, are specifically considered as the variables that model updates during the backpropagation phase; weights and biases are the core parameters of a deep neural network. Early in the training, bias is large, and variance is very small, whereas bias is small, and variance is high later in the training. If training is too long, the network will also have learned the noise specific to that dataset and is referred as overtraining. The minimum total error occurs when the sum of the bias and variance are minimal. Parameters are learned by the model during the training time.
Hyperparameters are the variable's settings that technically control the behavior of a network model by determining its structure and the way that a model is learned. Hyperparameters are set before the training and are trained on validation dataset before the optimization techniques; so, these are not learned from the training dataset; Below are the examples of some influential hyperparameters:

   - **Learning rate**: It determines the way a model is trained; it quantifies the learning progress of a model that can be used to optimize its capacity. It specifically learns how quickly the gradient updates follow the gradient direction.
   - **No. of hidden units**: Hidden unit is the layer between the input and output layer that determines the structure of a network model. It is important to regulate the representation capacity of a model. Normally, more the number of the layers, higher accuracy is obtained but it can suffer from overfitting by even learning noises of the layers if no proper regularization techniques are applied. Whereas, underfitting can happen with a smaller number of hidden units.
   - **Number of epochs**: It is the number of times the whole training data is shown to the network while training.

- **Batch size**: It is the number of patterns shown to the network before the weights are updated; it optimizes the training of a network by defining how many patterns to read at a time and keep in memory.

Hyperparameters optimization: Optimization is a way of achieving best performance on training data by making an adjustment in the model. Hyperparameters selection is the fundamental task to achieve high performance of a model. There is no hard and fast rule for its exact determination, but various manual selection methods and automatic deep learning algorithms like Grid Search, Random Search, and Bayesian optimization, ease the process of certain hyperparameters definition based on different criteria like cost function, memory requirement, nature of the training data and possible reduction in the test errors.

5. **Regularization** It is a process of avoiding overfitting in a deep CNN by introducing additional parameters. Popular regularization techniques include Dropout, DropConnect and Weight Decay and common method of using them for reducing overfitting includes reducing the network´s size by reducing the number of learnable parameters thereby decreasing its memorizing capacity. Another technique is adding weight regularization by making the network´s weights small and regularly distributed through the addition of cost to the loss function; it includes weight decay with L1 and L2.

   - L1 regularization includes the addition of cost proportional to the absolute value of the weight coefficient, whereas L2 regularization includes the addition of cost proportional to the square of the weight coefficients value. L2 regularization is also called weight decay.
   - Dropout: It is a regularization technique to increase the generalizing power of a network model by avoiding overfitting and results in the increment of validation accuracy. It is more suitable on larger networks with higher chances of learning independent representations. As its name suggests, it drops out the number of output features from the layer during training.

6. **Other useful terminologies:**
   Batch Normalization is the layer added to normalize activations of the input features before passing it to the next layer such that it helps in reducing the number of epochs for training network, prevents overfitting and stabilizes training process.
   Overfitting: It is the condition of learning to memorize the features on training data perfectly such that it performs properly on training data but executes bad performances on test data.
   Classification is defined as a supervised learning process having predefined classes of data fed as an input training data to result in the same classes as output on the untrained test data.

Learning Process: Training data is fed into the network through input neural network layers, passes through different hidden layers and output comes from the final fully connected layers in the same ways as the input supervised data; the result is compared among the trained and predicted data with discrepancy between them referred as errors. The network tries to reduce this error by changing weights of neurons in every iteration through back propagation mechanism and this process is called stochastic gradient descent (SGD). The parameter that determines the changes in weights is Learning rate. Training is the process of learning features of data through decision function.

7. **Transfer Learning** It involves the transfer of previously used models from which the new model can initiate a learning process on the new dataset based on the already learned features or patterns achieved from another dataset for other issues. Mathematically, Pan and Yang, 2010 has defined it as :
Given a source domain DS and learning task TS, a target domain DT and learning task TT, transfer learning aims to help improve the learning of the target predictive function fT(·) in DT using the knowledge in DS and TS, where DS ≠ DT or TS ≠ TT.
The transfer in this process can involve either the instant transfer of weights, or the transfer of feature properties, or the parameters used in extracting or training a model or the transfer of relational knowledge among two different data sources (Pan and Yang, 2010). In deep learning, transfer learning can be applied through feature extraction or fine tuning techniques as described below:

   a) Pretrained model as a feature extractor
   This strategy involves the use of pretrained model by removing the last fully-connected layer of the source data to extract features of the new data. Then, a classifier, either as a new fully connected CNN or machine learning classifier like linear Support Vector Machine or kNN can be added and trained on the extracted feature data as shown in Figure 2.2.

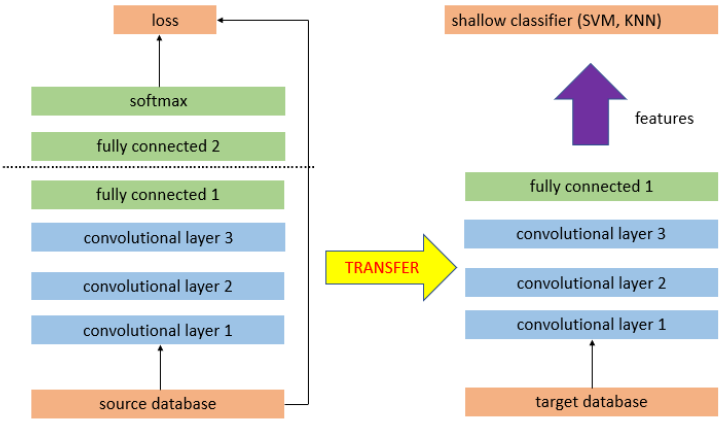


Figure 2.2: Transfer Learning Techniques as a feature extractor[5]

b) Fine-tuning the pretrained models

This strategy in addition to previous strategy involves either training all the layers and fine-tuning them by backpropagation or freezing some earlier layers and, fine-tuning together with back propagation on the rest of the layers (Yosinski et al., 2014). So, freezing layers involve no backpropagation updates, whereas fine-tuning the layers updates backpropagation. Normally, freezing is carried out to avoid overfitting by fixing weights on the initial layers as these involve more general features observation like shapes, edge and corners of the images. Also, learning rates (LR) can be varied during freezing and fine-tuning. Figure 2.3 shows a pictorial representation of finetuning a pre-trained model.

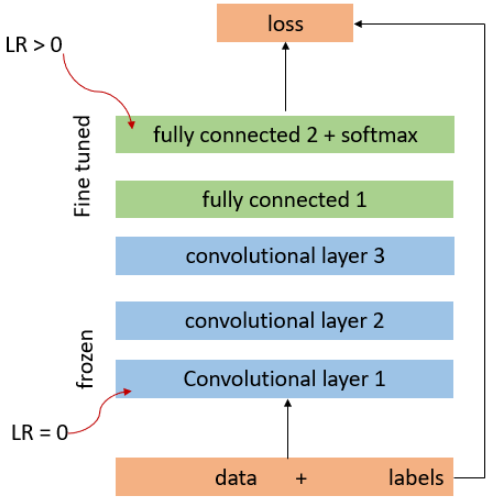


Figure 2.3: Transfer Learning Technique: Frozen and fine tuned layers, Source[6]

The essence of applying transfer learning can be described in terms of amount of data available and similarity or differences in the nature of that data. If a target data is similar to the previously used data, but is small, TL in CNNs can act as a feature extractor and linear classifiers can be used for final classification; if there is the availability of large amount of target data having high similarity with source data, the new network models can leverage the concept of fine tuning the entire network. Whereas, higher differences in data structure between the source and target data along with the availability of less data can use the concept of freezing initial layers and training the later layers; in case of large amount of data with high differences, weights from the previously trained models can be used to start training the new model (in accordance with [7]). The learnings from one model can be used to bolster the learnings from another model thereby

[7] "Stanford University CS231n: Convolutional Neural Networks for Visual Recognition"

compensating the lack of data, weights or parameters for improving the accuracy and model´s performances.

CHAPTER 3

DESCRIPTION OF DATA AND RESOURCES USED

This chapter consists of description about data source in its first section followed by a detail explanation on how data are prepared for this research in the second section. The third section presents a brief overview on the resources, including hardware and software used for the experimental set up of this research. The final section explores these data statistically and visually.

3.1 Data Description

The research uses "multi-camera multi-spectrum" airborne image sequences from Seagull Dataset[1] provided by the Seagull Project aimed for the research on maritime monitoring and surveillance. This dataset is maintained by Ricardo Ribeiro under VisLab-Computer and Robot Vision Laboratory of Instituto Superior Tecnico, Lisbon. The fixed-wing UAV named "Alfa Extended", mounted with cameras having electro-optical sensors operating in the visible spectrum, particularly LWIR camera receiving visible spectrum radiation and NIR, and a hyperspectral camera sensitive to radiation in the NIR and visible spectrum with the resolution of 1024*768 pixels, was used to capture videos over the Atlantic Ocean. It has a gas engine with 3.5 meters of wingspan and 25 kg of take-off weight; the payload carrying capacity is 10kg with the continuous flight of 8 hours. It was designed and operated by the Portuguese Air Force Research Center(Ribeiro et al., 2017).
The dataset is available in video format and consists of objects like cargo ships, smaller boats, sailing yachts, life rafts, dinghies, and hydrocarbon slick (fish oil spills, often

[1] http://vislab.isr.ist.utl.pt/seagull-dataset/

called simulating pollutants in the sea). Table 3.1 shows the status of raw data used in this study from the original dataset:

No. of Video	Type of Camera Sensor	Camera Model	Location
15	Visible and Infra Red	GoPro, Jai and Gobi	Portimao and Santa Cruz

Table 3.1: Data Description

3.2 Data Preparation

The data available in the form of video is first converted into image sequences using open source video editing software called FFmpeg with its default parameter of 25 for frame rate and the image size of 640*360 (Width = 640, height = 360). Since the research aims in recognizing the category of ship present in the images, all these videos were explored manually to identify the different types of ships available in them. All the images were divided into five categories based on the type of ship they contained; for the small boat, the number of images present were large and patrol boat though is a type of small boat was distinguishable from other small boats. So, patrol boat is categorized as new class to avoid data monopoly to the possible extent. Out of 15 videos, only one video contained Cargo, Dinghies were present in 3 videos, Patrol boat were present in 8 videos and small boat were present in 4 videos. While generating images from videos, there were the image sequences containing not a single ships and are categorized are the class with no ship. Since no of videos containing different classes of ships were varying, so is the number of images are differing in all the classes. The images with patrol boat and without any ship are present in large number compared to rest of the classes resulting in irregular distribution. Table 3.2 shows number of images available in each class and the definition considered during visual identification of each class by the expert.

In total, there are 33, 714 images consisting of ships and 13, 995 images without ships. The entire classified data is then divided into three datasets required for the process of training the dataset, validating it to optimize the accuracy and finally for testing the data to see if the final network model is correctly classifying the images and recognizing the object properly; test data are further divided into two subcategories: one for evaluation that requires similar supervised data arrangement, and another is the prediction dataset with no classes for the random prediction/recognition purpose. All images were randomly chosen for each of the dataset and data distribution is shown in Table 3.3.

S.No.	Name of Class	Number of Images	Definition
1	Cargo	1276	Class of images with ships that are more than 90 m long
2	Dinghies	806	Class of images with very small (less than 10m and no cabin) boats
3	Small Boat	4345	Class of images with ships that are less than 90m long
4	Patrol Boat	27287	Class of images with navy patrol boats that are approximately 27 m long
5	No Ship	13995	Class of images without any ship
	Total	**47709**	

Table 3.2: Data Preparation with image classes

S.No.	Data type	Ratio (of each class on the entire dataset)
1	Training	60
2	Validation	10
3	Test dataset: Evaluation on classified data Prediction on random image	 10 20
	Total	100

Table 3.3: Data Preparation with image classes

3.3 Resources Used

The research is carried out using open source software and packages. The mostly used platform are highlighted here.

Python is an object-oriented, interactive and open-source high-level programming language that consists of modules, classes, libraries and interfaces for easy programming in different operating systems with dynamic typing. The project uses Python 3.6 version installed in Anaconda environment.

Tensorflow is a Google Brain team created open-source library offering high computational tasks like machine learning and deep learning through usages of computational platforms like CPUs, GPUs, and TPUs. It uses Python as a front-end API. The research uses Tensorflow 1.12 version for the project.

Keras is a user-friendly, high-level python written API that supports easy and faster deep learning neural networks with minimum coding. It is developed by François Chollet from Google and run on the top of Tensorflow, Microsoft Cognitive Toolkit

(CNTK) and Theano. It offers a complete framework for successfully building and running a neural network model. The research is using Keras 2.2.4 version.
Within these main frameworks and programming platforms, other modules, libraries, and packages supported by python were used for image visualization, developing deep CNN networks and plots. These include numpy, pandas, scikit-learn, opencv, and matplotlib. In terms of hardware, the research uses NVIDIA GTX 1080 GPU with 11 GB of RAM capacity on a server setting with linux operating system.

3.4 Data Exploration

The prepared dataset is then explored through a bar diagram 3.1 to see the presence of images in each category and is then visualized to see the quality of images. It is clearly

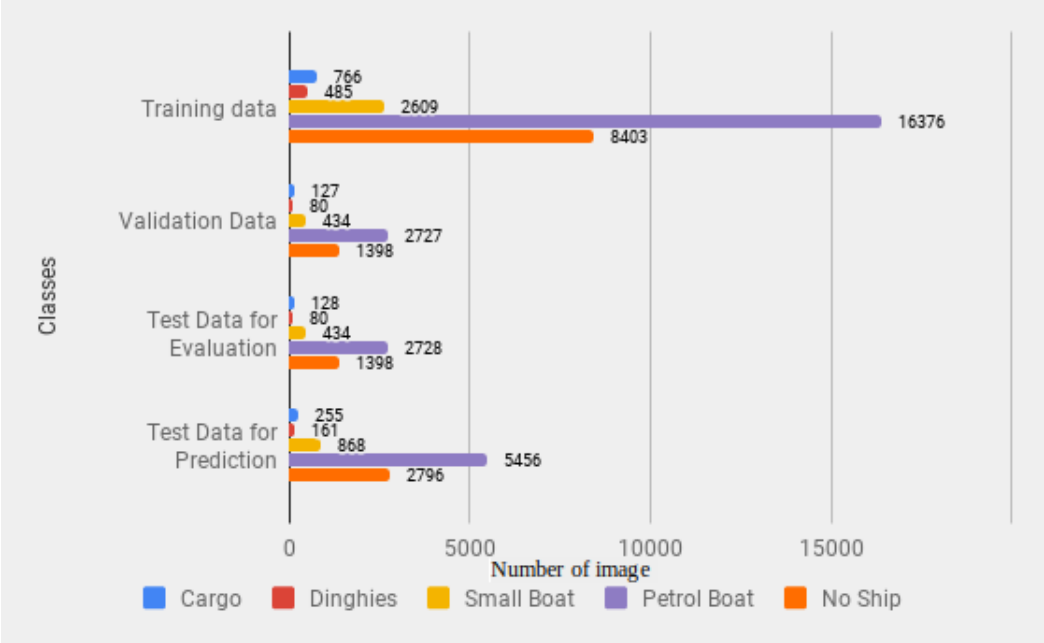


Figure 3.1: Data Distribution for Training, Validation and Testing Purpose

observed that in each of the classes, the highest possession is for patrol boat and lowest possession is for Dinghies.
Likewise, further visual exploration was done on the images to see their quality. Images contain many variations even within a class based on the video from which they were generated.Figure 3.2 displays variability present in some of the images used for study.

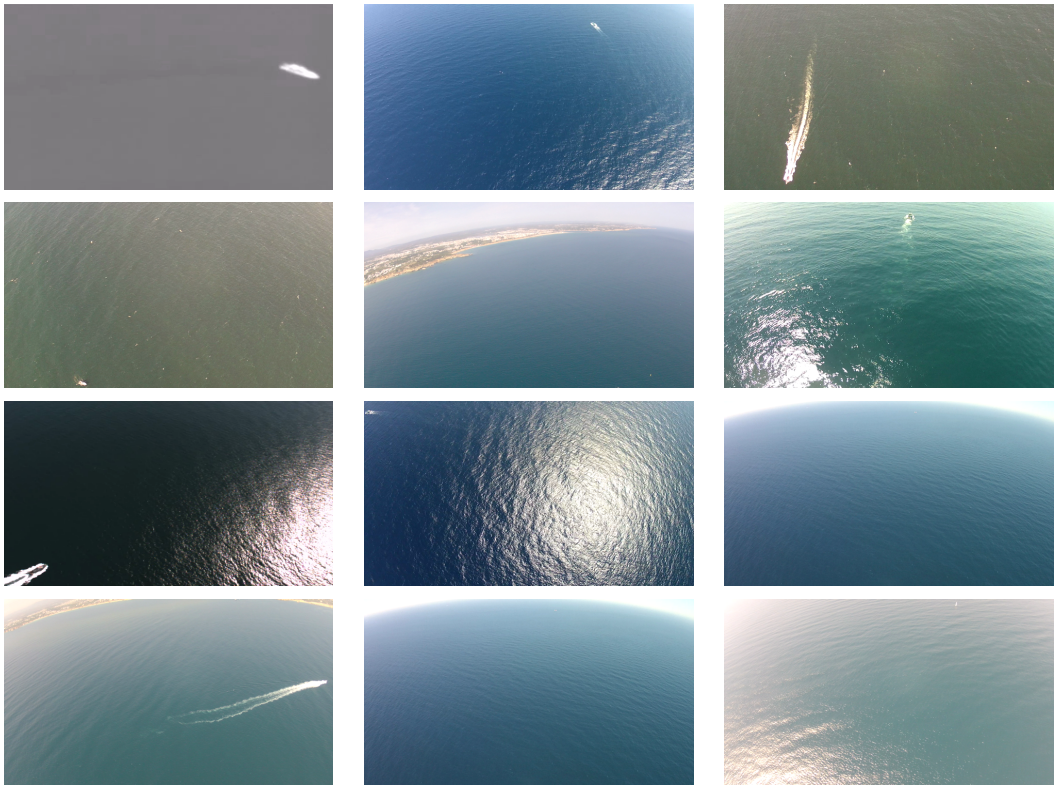


Figure 3.2: Data Exploration: images with varying background environments

CHAPTER 4

METHODOLOGICAL DESCRIPTION

The first section of this chapter describes the methods proposed for this thesis work and overall conceptual mechanism. The second section consists of the modification of chosen architectures and proposes new architectures along with fine-tuning and hyperparameters setting. The third section is related to the implementation aspects of the methodology that describes algorithm designs for visualizing output of the models designed, and presented with customization to train on the thesis dataset.

4.1 Proposed Methodology

Based on literature reviews done on the existing techniques and state of art performances demonstrated by machine learning and deep learning technique for image recognition, this method proposes the use of deep CNNs for identifying and classifying ships in the thesis dataset. Training the thesis data on a completely new architecture within the available timeframe and resources may not be a good option to achieve higher accuracy as the number of available images are not sufficient enough with irregular quantity of images present in different classes of ship and the challenges associated with their varying background environments. So, this study proposes the use of freely available state-of-art outperforming architectures, namely VGG16, Xception, ResNet50 and InceptionResNetV3. Also, this thesis puts forward the idea of using transfer learning techniques on the previously trained network models to cope with the quantitative issues of images and to harness the beneficial aspects in terms of parameters and weights sharing while using less computational time and resources. Multi-classification with 5 classes of ships will be carried out on the models pre-trained on ImageNet dataset using above mentioned architecture using transfer learning strategies and hyperparameters optimization. The entire process of building a model and

recognizing the class of ship involves model construction, training, evaluation and prediction that is proposed to initialize by following the three steps mentioned below to be executed using high-level programming of keras with tensorflow at the backend in Anaconda-python environment.

- Selecting the pre-trained models
- Design and modification of network architecture
- Optimization and regularization with Hyperparameters

1. **Selecting pre-trained models** Four pre-trained models, all showing state of art performances during their release are chosen considering their network accuracy, performances and associated network complexity at the time of release; they are VGG16, Xception, ResNet50 and InceptionResNetV2. VGG 16 though is a relatively shallow architecture than others was the deeper CNN network when it was developed and offers simple network complexity with higher accuracy. Xception has been released as the improved model architecture of different versions of Inception models and hence represents their advancement. ResNet50 introduced the concept of residual networks offering more deeper CNNs and InceptionResNetV3 is the integration of concept from both the inception and ResNet as its name suggests. All the chosen network models are pre-trained on ImageNet dataset and are available with keras API. The ImageNet dataset has remarkable number of images containing different types of ships that are available in its database through different names like ship, boat, vessels, cargo. Table 4.1 shows accuracy obtained on the chosen models generated by training on ImageNet dataset.

Model	Top-1 Accuracy	Top-5 Accuracy	No. of Parameter	Size
Xception	79	94.5	22910480	88 MB
VGG16	71.3	90.1	138357544	528MB
ResNet50	74.9	92.1	25636712	98MB
InceptionResNetV2	80.3	95.3	55873736	215MB

Table 4.1: Accuracy of the models chosen in the study when trained on ImageNet dataset (Accuracy is expressed in terms of %correction) (Source: Chollet, 2015)

2. **Design and modification of chosen network architecture**
Both the transfer learning techniques of using CNNs as feature extractor followed by a classifier and fine-tuning with their previously described strategies will be used for all the pre-trained models on the new target data of this research.
3. **Optimization and regularization with hyperparameters**
It involves changing the parameters of optimization and regularization techniques that best suit conditions of the above strategies. Hyperparameters are changed simultaneously while applying step 2 and the models are trained with different values. The

hyperparameters strengthening the model and fitting it without under and overfitting with the highest accuracy possible will be chosen through heat and trial method.
The resulting models will be the modifications of existing pre-trained models suiting the new classification and recognition scenarios of the research dataset. All the models will be compared based on the classifiers chosen, layers frozen and trained along with the layers complexity, parameters used for training the dataset, hyperparameters used and time required for training the target data sets. Additionally, their performances will be compared in terms of training, validation and evaluation accuracy. The model showing best performances on all these criteria will be recommended as the final model for prediction on the unseen datasets under similar environment.

4.2 Methodological Design and Implementation

4.2.1 Data Preprocessing and Augmentation

Since the method aims to use deep learning techniques, particularly deep convolutional networks that are supposed to learn features of the supplied images by extracting their properties and using them for future prediction, this study is applying less preprocessing possible to provide real world scenarios associated with the image datasets; preprocessing techniques applied to all the data includes: rescaling the images in the range of 0-1 to ease image processing system for feature attraction with the small range variation prepared by multiplying with the factor of $1/255$ and image resizing as different deep CNNs described below demand different input size. Both of these steps are performed within deep learning environment by avoiding any work outside of it.

4.2.2 Modification and design of pre-trained Network model Architecture

The last fully-connected layers are removed from the network architecture model; since the research dataset consists of only 5 categories, the last dense layer will have 5 classes instead of 1000 classes from default ImageNet dataset. The classification work is then carried out by using FCNNs dense layer having softmax classifiers after applying necessary modifications or directly after extracting the features.
The next way of modifying and developing models is through fine-tuning techniques: some of the earlier layers on the network are frozen for the generic properties they learn, and rest of the layers responsible for extracting target data specific high level features are trained on the dataset. The numbers of layers frozen for different layers are varying as different networks have different layers and structures:heat and trial method is used by observing the accuracy while changing the numbers for frozen layers. Below is the description on proposed techniques adopted for each of the network models with modifications in their architecture.

- The VGG16 pre-trained on ImageNet dataset in default settings consists of 5 blocks with 13 convolutional layers and the last classification layer consisting of a flattening layer, two fully-connected layers followed by the final prediction layer. The first two blocks have 2 convolutional layers and the rest have 3 convolutional layers in each of them; all the blocks contain MaxPooling layer at their end. The modification and design of new model is done in different ways, either by freezing all the layers except last four layers or freezing all the layers and adding different dense layers to experiment the modified models with different hyperparameters, activation function and regularization techniques while training on the research dataset. Section I.1 of ANNEX I shows the overview of proposed changes and original setting of VGG 16 structure with its parameters.
- The original ResNet 50 pre-trained on ImageNet dataset consists of 5 blocks and 2 top layers together making 176 layers in total; each of the block has 3 deep layers with different CNN layers, activation layers and batch normalization layer. The topmost layer consists of a GlobalAverage Pooling layer and a fully-connected layer with 1000 classes. Alteration in its architecture mostly will include training of layers above 163 and freezing all the earlier layers; different hyperparameters optimised and regularized with different values will be applied for achieving acceptable accuracy. It has a complex structure and accepts input images of size 299*299.
- Xception consists of 14 blocks and 3 top layers that makes a total of 133 layers in the default settings of pre-trained Xception model on ImageNet data set. Each of the block consists of different CNN layers followed by batch normalization and activation function. The topmost layers include an activation layer, average pooling layer and the final prediction layer. The new architecture modified from the original will include frozen layers prior to 126 and the rest are trained together with different hyperparameters. It accepts input image of size 299*299.
- InceptionResNetV2 has 781 layers constituted in 20 concatenated blocks and 2 top layer in its default settings pre-trained on ImageNet data set. Each of the block consists of different CNN layers followed by batch normalization, activation function, concatenated and lambda layers. The topmost layers include an activation layer, Global average pooling layer and the final prediction layer. The new architecture modified from the original will include the frozen layers prior to 777 and rest of the layers will be trained with new settings. It accepts input image of size 299*299.

Table 4.2 shows the overview of modification applied in the pre-trained models. The original structure, blocks and associated parameters of ResNet50, InterceptionResNetV2 and Xception are provided at the github[1] link considering the long and complex

[1] https://github.com/Laxmi15/Ship-Detection.git

structure of these models and limitation with this report format.

Changes in Layers	Type of Pooling layer	Activation Function	pre-trained model	Frozen layers (all the layers before:)
Addition of a dense layer (1024) and a Dropout layer	Global Average and Global Max	ReLU and ELU	Xception	All, 126 and 129
Addition of 2 layers (1024) and two Dropout layers			VGG16	All, 3, 4 and 5
Addition of 2 layers (1024 and 512) and two Dropout layers			ResNet	All, 163 and 166
Addition of 3 layers (2 layers of 1024 and the third layer of 512 and two Dropout layers			InceptionResNetV2	All and 777

Table 4.2: Modification strategies applied in the layers of existing network models

4.2.3 Optimization and Regularization with Hyperparameters

The main hyperparameters proposed to train the models to achieve better performances of the deep CNNs model through model optimization and regularization are listed in the Table 4.3 with corresponding values. Based on the literature reviews on different optimization and regularisation techniques for hyperparameters, major influential variables, namely learning rates, pooling layers, activation functions, batch size, regularisers, batch normalization and dropout are experimented with different experimental set ups. Dropout is tested with the usually considered standard value of 0.5 (Srivastava et al., 2014) and no of epochs is kept 20 based on the available hardware resources with the provision of early stopping if the validation accuracy is not increasing for certain no of epochs.

Variables	Values
Learning rates	1e-4/5 and 2e-4/5,
Pooling Layer	GlobalMax and Global Average
Activation	ELU and ReLU
Batch Size	8, 16 and 32
Regularizers	L1 and L2

Table 4.3: Optimization and Regularization with Hyperparameters

4.2.4 Algorithm Design

Four main algorithms are designed for four of the chosen pre-trained models following their network architecture in two ways: Sequential and functional API. VGG16, being comparatively shallow and simple structure, is trained using Sequential and rest of the network models are trained using functional API. These algorithms are compiled and programmed following the freely available resources available at Keras website[2] to suit the research scenario. Other algorithms are only the variations in number of layers and hyperparameters mentioned in Table 4.2 and Table 4.3. All the algorithms used in

[2]https://keras.io/

this research are made available at github link [3] and one sample algorithm is attached at the ANNEX III. Here is the brief description on the overall working mechanism of VGG16 algorithm mentioned at the Annex III. The basic python modules for handling data, plotting figures, using Keras applications and related libraries were imported initially. These include numpy, pandas, sklearn, Keras modules for pre-trained models and related applications. To address variations associated with the data mentioned in the Data Preparation Section, data was prepared with manual inspection to ensure proper distribution of data in all the dataset and no data split task was done while feeding the data. Instead, an image data generator class was used to load images from concerned directories as a real-time data feeding; it has different parameters that can be applied for both the preprocessing and augmentation techniques as per the necessity of research data. Data augmentation is applied since the research dataset consists of fewer images in the classes like cargo and dinghies. This technique changes the pattern of images based on parameters used for providing variability in the images and transforms them such that each image is trained only once. So, augmentation is only applied to the training data through data generator and helps in generalising the model properly. However, augmentation demands more computational resources and time so experiments have been done with and without augmentation to see if they are influencing in the validation accuracy. For each of the generator, target size is defined based upon the input size requirement of the network architecture to be used. Batch size is assigned normally as the power of two depending upon the capacity of RAM. The Class mode is binary for two classes and categorical for multi-classes. Shuffle is applied only to training dataset to arrange the images randomly such that no image is read twice.The algorithm is similar for all the models till data preparation and importing of the models pre-trained on ImageNet dataset but differs on the way the models are modified afterwards. autorefann:annex1 shows the import for VGG 16 but other architecture like inceptionv3, resnet50 can be imported in the similar way by calling respective model´s name. The top fully-connected layer is removed from the network model to replace it with our classifier as the default in Imagenet contains 1000 classes whereas research dataset contain only 5 classes. Input shape is the size of images and it's RGB channels. The model summary can be displayed; it helps in deciding the layer that are to be trained and the layers that are to be frozen by observing their positions and associated parameters in the network´s architecture. If the layers are frozen, the weights will be fixed and will not be updated while adding new layers, models and training in every epoch. Normally, initial layers are frozen as they extract generic features like edge, geometry and latter layers are trained as these specifically extract high level features. Then the model can be created. As mentioned before, different approaches are applied for VGG16 and other models. There is no specific rule for how many layers are needed though more layers are considered to have higher

[3] https://github.com/Laxmi15/Ship-Detection.git

accuracy. Batch Normalization can be added usually after non-linearity to prevent internal covariate shift. Dropout and activation functions as described before can be changed. ReLU and eLU are experimented with the models proposed here. Except for creating the models, other processes are similar again with the Sequential VGG16 and functional API for other models. After this, the model is ready for compilation with various hyperparameters, loss function, optimizers, and learning rates. The metrics held accuracy and loss of the model.

The network is ready for training on the thesis dataset. As a good practice for not wasting computational resources if the validation accuracy is not improving after 10 epochs, checkpoints and early stopping can be set such that it would save model with the best accuracy. This strategy was exercised while training the thesis data. The loss and accuracy between training and validation data can be visualized through plots and overfitting can be checked. Depending upon the performances, we conducted various tests with different hyperparameters and the best fitting values were used for the final model. The saved model can be loaded later for evaluation and prediction on the test data. Keras also offers real-time testing without saving the model if it is not required for the future. Saving the entire model requires high storage depending upon the parameters settings and dataset trained; Validation data is used to optimize the model. Since it is already used in training the model, using the same data again for the evaluation will yield a good result as the model has already learned the features. So, completely new data should be used for evaluating the model´s accuracy. The evaluated test data can be predicted further with their respected class of ships using predict_generator, wrongly classified errors can be calculated and each of them can be viewed along with the prediction probability. Also, the randomly chosen test data with no classification can be predicted using the trained model and predict_generator. The code at the autorefann:annex1is used to identify such a randomly arranged test data with no classes in it and saved in csv format. Even individual images can be recognized using the trained model. Similarly, confusion matrix can be created to know true-false positives and accuracy-loss comparison along with the computation of precision, recall and f1-score. Individual codes for the final model created using each of the network architecture pre-trained on ImageNet and trained on thesis dataset are shared through gitbub link.

CHAPTER 5

RESULTS AND PERFORMANCE EVALUATION

This chapter is the visualization of what has been proposed, designed and programmed in Chapter 4. It´s first section consists of the output of methodological process associated with generating the modified models and their performances with different hyperparameters settings. The second section consists of evaluation expressed in terms of validation and evaluation accuracy, performance metrics and visual inspection.

5.1 Visualization of Data Augmentation Technique

Data augmentation techniques is applied with the rotation of 2 degree, width and height shift range of 0.2 with nearest interpolation, and horizontal flip. Figure 5.1 shows the output obtained with these parameters on the images used during the model training.

5.2 Modification and design of pre-trained Network model Architecture

Modification in the layers and trainable parameters:
Different models used in the thesis consists of different number of layers available in their architecture based on which this thesis has performed experiments for freezing and training the layers. The modification in the original architecture of all the models involves the addition of different layers like dense (convolutional) layers, pooling layers, activation, layers, batch normalization layers and the final dense layer with softmax classfiers having 5 classes for the classification of thesis dataset with the modified

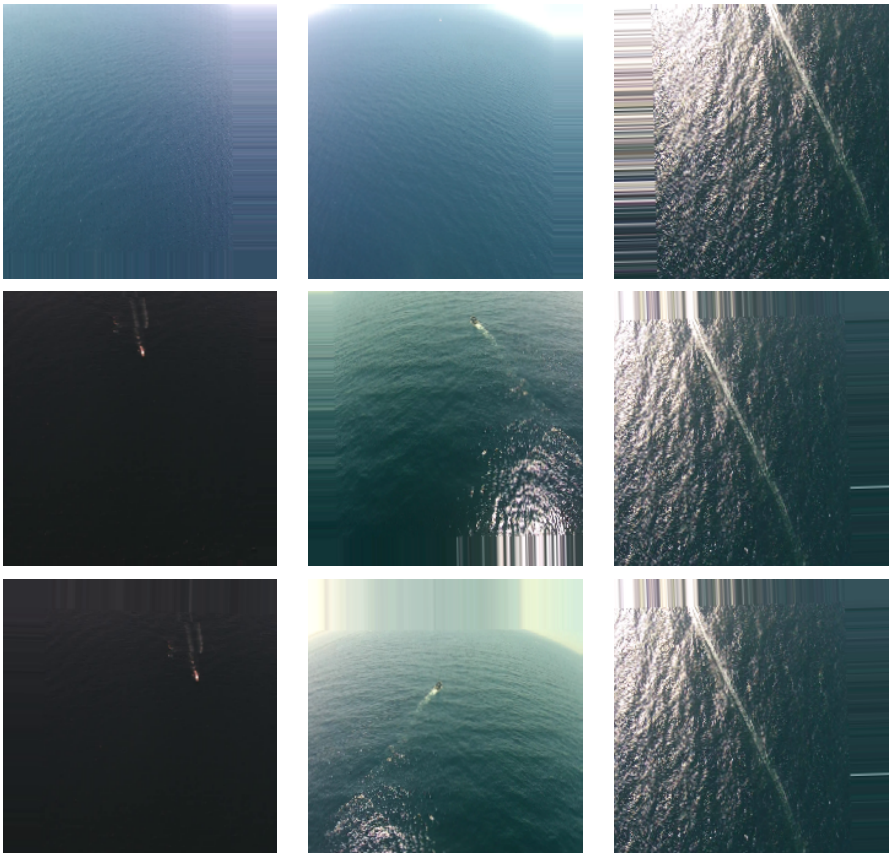


Figure 5.1: Images obtained with Data Augmentation Techniques

models. However, it has been observed that only the changes in convolutional layers and batch normalization are responsible for determining the number of trainable parameters to be used for training the models on the thesis dataset and the trainable parameters obtained for the modified models are either equal or greater than the original parameters. Table 5.1 provides the list of some of the representative models and their trainable parameters used in the thesis.

Model	Layers	Addition of a dense layer (1024) and a Dropout layer	Addition of 2 layers (1024) and two Dropout layers	Addition of 2 layers (1024 and 512) and two Dropout layers	Addition of 3 layers (2 layers of 1024 and the third layer of 512 and two Dropout layers
Xception Layer		22,910,253	23,959,853	23,432,493	24,482,093
VGG16		40,405,824	41,455,424	40,930,624	41,980,224
ResNet50		25,637,893	26,687,493	26,160,133	27,209,733
InceptionResNetV2		55,855,205	56,904,805	56,377,445	57,427,045

Table 5.1: Changes in the number of layers and corresponding number of parameters

Also, the number of layers frozen or trained does not influence the parameter determination. However it is important for the network´s training and validation accuracy as the frozen layers do not participate in backpropagation and weight updates, whereas all other trained layers are sharing the errors among them. Model´s performance is

also varying with the variation in number of frozen layers. In Xception model, adding 3 layers, each followed by pooling layers though was increasing the trainable parameters and all the layers before 126 were frozen, the model´s accuracy for training and testing was better with the layers frozen before 129; different experiments were done by changing the hyperparameters; layers when all frozen and the addition of a dense layer with 512 filter size followed by the dropout layer of 0.5 and the final classifying dense softmax layers of 5 classes performed with the highest accuracy (Figure 5.2). Similar trends were observed for ResNet50 when freezed with 163, 167 and all the layers in the base models were freezed with the addition of same layers like Xception. The InceptionResNetV2 experimented with two case: freezing all the layers and freezing layers before 777 showed the alike trend with Xception and RexNet50. In case of VGG16, performances were using both the techniques: freezing all layers except four of its last layers and training only the top layers added later by freezing all other VGG 16 layers, both showed high performance; the latter experimental setup follows the similar composition of layers like ResNet50, Xception and InceptionResNet. The graphical visualization of training and validation accuracy is available at ANNEX II. In terms of Pooling layers, all the models were trained with and without global average and global max pooling layers. Results can be observed at ANNEX II. The model without pooling layers were performing badly and both the global average and max pooling increased the model´s accuracy remarkably. However, clear distinction was observed between Global Average and Global Max pooling; global max pooling performed better for the similar hyperparameters and layers settings than the global average pooling for all the models.

Modification in layers and hyperparameters:
Hyperparameters as specified in 5.1 were highly responsible for the robust models with better performances but not with all the variables, values and environmental settings. Learning rate when applied with the value of 1e-04 though was bridging the overfitted gap between training and validation accuracy to some extent, it was not acceptable when used with the globalmax pooling followed by a dense layer of 1024, dropout layer of 0.5 and final dense-layered-softmax classifier as observed by the model´s performance in terms of evaluation and prediction on new data set. Comparatively, LR with the value of 2e-04 performed better with higher accuracy than the one observed with LR of 1e-04. However, the best accuracy observed with the model was obtained by using LR of 1e-05 with different settings. The ELU and ReLU were also experimented in different models but RelU performed better than ELU in both the cases of layers frozen at 126 and 129 for Xception model. The dropout values used as 0.5 when tested with the models always performed better; without dropout values, models though had higher training accuracy suffered overfitting to the large extent followed by bad evaluation. Epoch was set at 20 with early stopping patience level of 10 if the validation accuracy was no more increasing. Very few of the models as shown in

the Figure II completed whole 20 epochs; most of the layers started stopping at earlier epochs like 12, 13, 15, 16, 17, 18 and 19. L1 and L2 regularization were tested under the same environmental settings with the value of 0.001; L2 regularisation performed better than L1.
Batch size is experimented with the values of 1, 8, 16 and 32 and it is highly dependent on the computational capacity of hardware resources; so different batch sizes were used for different datasets considering its usefulness; since the training process needs to learn from the features with large datasets, experiments were done with 32 and 16 for the corresponding batch sizes of 16 and 8 in validation dataset respectively, whereas the test dataset was tested with batch size of 1 as it is not to be learnt by the model. Under the similar hyperparameters and layers experimented, the former setting described here with 32, 16 and 1 produced overfitting in the models as shown in Annex I and performed poor during prediction whereas the latter settings of 16, 8 and 1 formed highest performance in all the models with highest evaluation accuracy too. However, epoch per image sample was 749 when using 32 batch and it´s 1789 while using 32 batch size as it is the ratio of number of images and batch size used while creating the dataset. The training and validation accuracy graphs are provided at the section II for all the methods with different parameters described here. Among all the four models discussed here with different settings, the best performance setting shown by all of them are listed below:

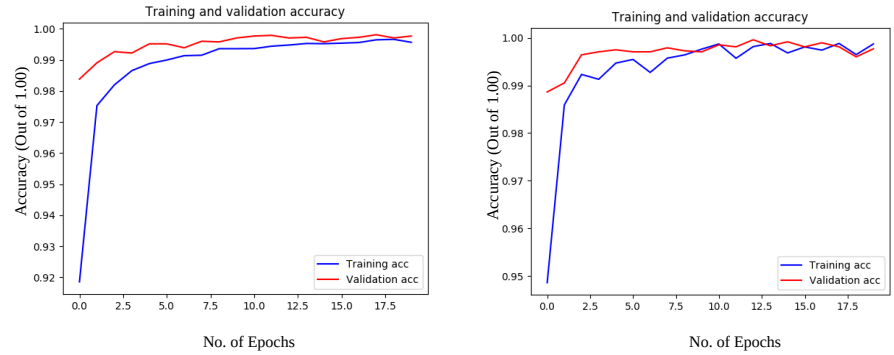
S. No.	Method	Layers Composition	No of trainable parameters	Time for model (sec)
1	InceptionResNetV2	base model+Global	55,065,701	21346.77
2	ResNet	Max Pooling Layer	24,586,245	9454.87
3	VGG16	+Dense Layer of	14,840,133	6182.21
4	Xception	512 with ReLU	21,858,605	15475.59
		+Dense Layer with Softmax		

Table 5.2: Models and their structure

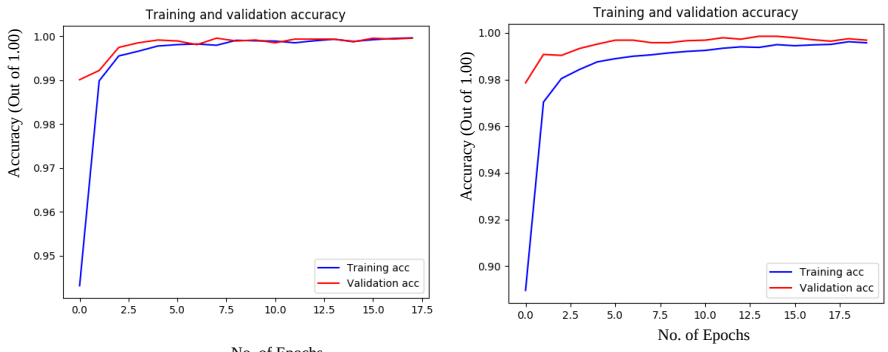
5.3 Performance evaluation and comparison

The best performing models from all the network architecture used in this thesis have very competitive and high training-validation accuracy as observed in Figure 5.2. To evaluate if the models are performing well, all of them were then tested with the unseen test dataset having same number of classes. Figure 5.4 shows the matrices of all the models and their corresponding evaluation accuracy are shown in the Table 5.3.

High accuracy obtained during the model training was well justified with the evaluation accuracy and predicting errors obtained for each of the model. Also, precision

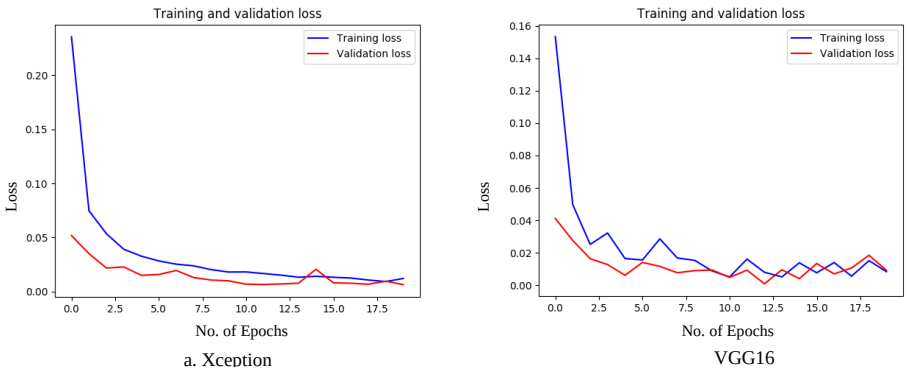


a. Xception b. VGG16

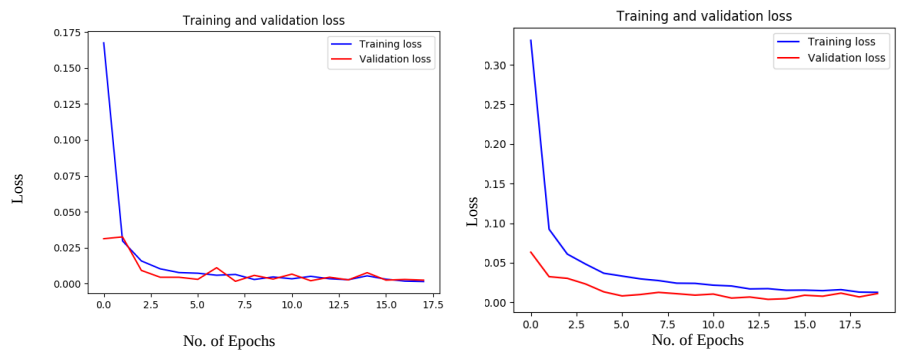


c. ResNet50 d. InceptionResNetV2

Figure 5.2: Training and Validation Accuracy of the chosen models



a. Xception VGG16



c. ResNet50 d. InceptionResNetV2

Figure 5.3: Training and Validation Loss of the chosen models

S.No.	Method	No. of Epoch	Batch Size	Evaluation Accuracy	No of errors
1	InceptionResNetV2	20	16 on Training	99.769	11
2	ResNet	18	and 8 on	99.900	0
3	VGG16	20	Validation	99.660	17
4	Xception	20	Dataset	99.937	3

Table 5.3: Comparison of Evaluation accuracy between different methods

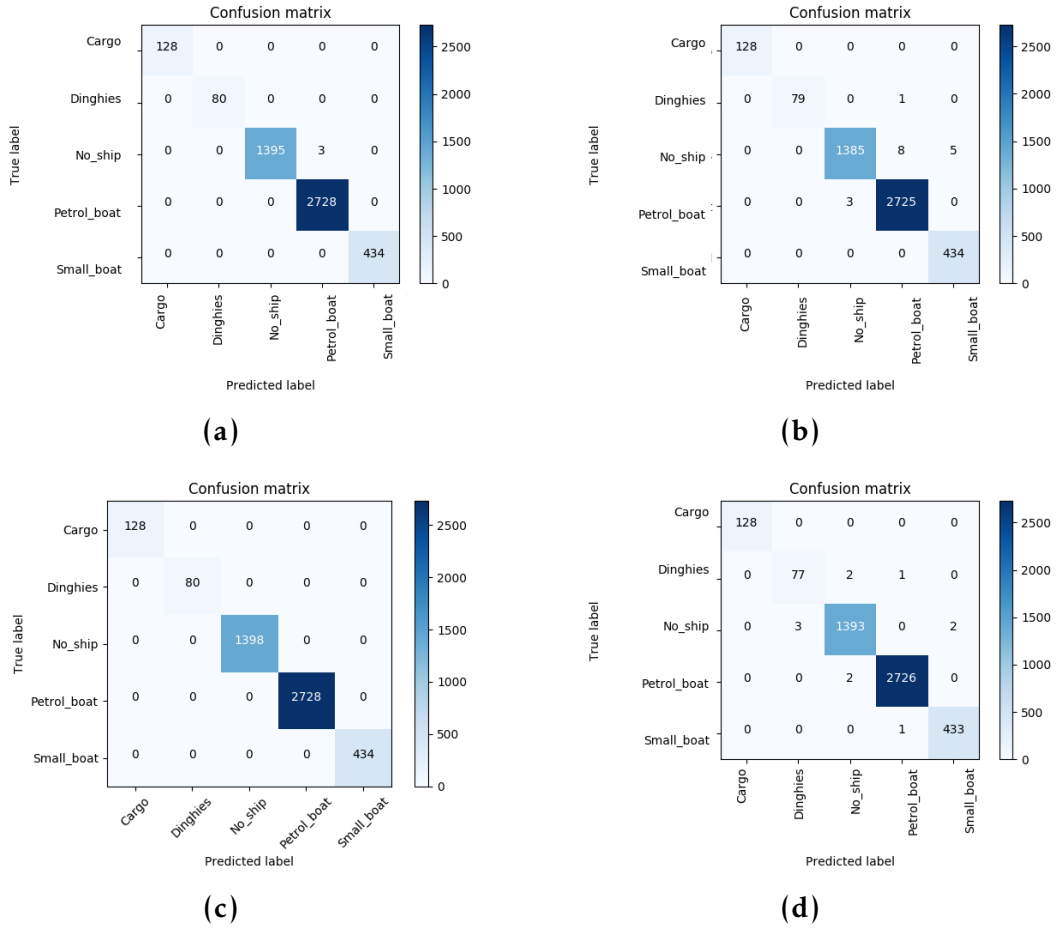


Figure 5.4: Confusion Matrix of (a)Xception, (b)VGG16, (c) ResNet50 and (d)InterceptionResNetV2

and recall calculated on the test data were 1.00 for all the final models. Since the models were showing full precision rate, the randomly prepared test dataset with no classes on it was used for further testing the model performance through recognition of the class of ship in its images; all those prediction were manually inspected and the model was consistent with its accuracy and predicted the ship classes correctly. The algorithm also allows visualization of wrongly classified images and can also be used to recognize the new class of ship present in an image as below:

Figure 5.5: Visualizing images with the ship classes using modified (a) Wrongly classified image and (b) Ship Recognition on new image

Chapter 6

RESULTS DISCUSSION AND COMPARISON

This chapter consists of analysis and discussion on the output obtained in Chapter 5 and then compares the outputs with similar works done in other research.

6.1 Result Discussion

The results show that all the models have low training and higher validation accuracy at the beginning as depicted in the Figure 5.2 and and exactly opposite trend for loss as observed in 5.3. This could be because of bad image quality caused by the presence of noises like sun glares, tides and waves available in the original research dataset. It is observed that learnings and the predicting capability of the model are highly influenced by the hyperparameters. Among different variables experimented, it is observed that the learning rate should be small in the beginning of model training. The ReLU activation function performs better than ELU and dropout is very useful in reducing the overfitting between training and validation data. Among L1 and L2 regularisation, L2 performed better than L1 but in case of this research, model performed well without using regularisation techniques as well. Applying Batch normalization shows robustness to some extent by slight reduction in the computational time but also caused drop in training and validation accuracy. In terms of modification in layers, changes in convolutional layers are responsible for determining model's parameter. Globalmax pooling performs better than the global average pooling.
In terms of pretrained models used, VGG16 when trained on last four models by freezing rest of the models, though possessed high training and validation accuracy performs badly by predicting 147 images with wrong classes as shown by the Figure **??**. Other models showed both the validation and prediction accuracy less while freezing

some of the layers. The model offering highest accuracy was by freezing all the original layers and addition of new layers. VGG16 though has fewer convolutional layers and less network complexity, it´s performance was competitive with other models and required less time for model training; however, the figure 5.2 and 5.3 shows some irregularities in accuracy/loss plots that might results in poor model performances and also represents that model is not learning consistently.
InceptionResNetV2 with the highest number of parameters required longer training duration compared to all other models but its learning is gradually improving as the epoch number increases. Xception model shows similar accuracy-loss trends in the starting and ending but has irrgelarity during epoch 13-14. ResNet50 shows early fitting with a minor discrepancy at epoch 6 followed by acceptable fitting afterwards. . Also, it required lesser training time than Xception and InceptionResNetV2 but more than the VGG 16 required. In terms of evaluation performances, all the models have very competitive accuracy with slight differences in the number of errors on the classes of ships predicted by them. The precision and recall of 1.00 was obtained. The confusion matrices at Figure 5.4 shows that Xception recognized 3 of the ships wrongly, VGG16 recognized 17 ships wrongly ResNet50 recognised all the ships correctly and InceptionResNet recognised 11 ships wrongly. Considering the slightly higher training time consumed by the model than VGG16 but lesser than other models, the effective model´s performances with both the validation and test dataset, ResNet50 is preferred as the robust model than others for the ship recognition tasks carried out in this research.
The results show that pretrained are useful to address the problems of less resources for the tasks like image recognition. It is not necessary that having higher number of parameters and more number of layers performs best than the model with less parameters. Also, it shows that hyperparameters with proper optimisation and regularization can help in improving the model´s performance greatly. All the models achieved their best accuracy when following the experimental setup with final configuration mentioned at Table 5.2; the performances shown by all the models are very high. These models were configured with different settings and evaluated with both the supervised classes of data and random classes of data but the performance remains high every time. The major change with these models variables leading to highest accuracy was depicted clearly by the variation in batch size; models were having less accuracy and suffering overfitting problems when the training batch size was 32 but the accuracy was increased to the full capacity when the same settings were changed only reducing training batch size to 16. Batch size referred in the study is functioning as mini-batch in the model and represents the number of updates while training the model. It follows the concept of Stochastic gradient descent and smaller the minimatch is, more is the updates in the model´s weights. So, more updates mights have resulted in obtaining the better weights that ultimately caused rise in accuracy. It´s lower value consumes

more time for model training and increases the number of epoch per images thereby increasing epoch duration such that the model can backpropagate frequently to reduce the errors. Another possible reason could be the presence of closely related image sequences causing much similarity between the training, validation and evaluation dataset though random selection was done while preparing the data and additional evaluation was done as the ship recognition task on radomly distributed images with no class in them. Also, the models involve image augmentation tehnique which might have increased variability in the images thereby increasing the model´s performances with the different background environments present in the images.

6.2 Comparison with existing works

Further exploration was done with the existing works involving similar techniques for ship recognition to compare their similarities, differences and performances. Earlier works on Seagull dataset include Cruz and Bernardino, 2016's research for boat detection using CNN-pretrained models on ImageNet; they used sliding window and salient candidate regions for detection and CNNs architecture for classifying the object detected as class boat and not boat, particularly using AlexNet and GoogleNet. The comparison done in terms of recall and precision shows 99.4% precision for the recall of 50%; their research also obtained higher precision of near 100 percentage for object detection. No research on the multi-classification of a boat on this dataset has been available until this research has been conducted. However, some research on ship classification using pretrained network models on different datasets are found. Gallego et al., 2018 used transfer learning appraoch: VGG16/19, ResNet, Inception V3 and Xception for extracting feature with CNN and then performed classification with kNN; with Inception, they obtained 99% accuracy on ship classification by outperforming the earlier achieved 79% with traditional methods. Instead of existing top layers, they used Global Average Pooling, Fully-connected Layers with ReLU and Dropout value of 0.2 with the output size of 2048 and 1256 with softmax layer for classification; accuracy using softmax classifier was 98.02 and kNN resulted in 99.05% of accuracy. They evaluated the developed method to classify ships on MWPUVHR-10 data and outperformed the accuracy obtained with existing state-of-art methods. Compared to this method, we used Global Max Pooling, dropout value of 0.5 and learning rate of 1e-05 thereby obtaining the accuracy greater than 99%.
Leclerc et al., 2018 used Inception and ResNet architecture on Maritime Vessel dataset to classify ships for target tracking; they replaced the last FC layers and changed the number of layers, value of L2 regularisation and learning rates together with the fixed mini batch size of 128, momentum of 0.9 and 300 epochs. The highest accuracy of 78.73% was obtained with InceptionV3 using L2 value of 0.0005. Our method though experimented with L2 value of 0.002, the final models obtained remarkable accuracy without using L2.

Miličević et al., 2018 used transfer learning technique with VGG19, InceptionV3, Xception and ResNet50 on limited dataset on MARVEL dataset for ship classification with data augmentation: parameters they generated consists of the epoch value of 200, lr rate of 0.00001 with RMSProp optimizer and mini-batch size of 64 generated through Grid Search method. Though their training accuracy was greater than 99%, overfit prevailed in the model limited the validation accuracy of 76% with VGG19 as the highest among the methods used. This study used Grid Search, which by nature is popular for producing highest accuracy regardless of how the model will perform later. Our study used heat and trial method instead of existing grid search and random method to avoid such performances, and also to understand the influence of each hyperparameter in the models.

If we look at the structure of these models, these are trained for more number of epochs with higher mini-batch size and larger output size. Though the models can not be compared with each other as they differ in their training dataset, our models consist of comparatively less complexity and high accuracy than other models trained on each of their own dataset. Existing research works show that accuracy can be drastically high using transfer learning approaches with the deep learning techniques. Our models can be further evaluated with different datasets like as Gallego et al., 2018 did with the MWPUVHR-10 dataset to ensure its practical usability in other environment. Also, existing machine learning techniques like SVM or kNN can be applied as classifiers to compare the accuracy on similar dataset with different approaches. However time constraints together with available computing resources to perform these tasks at the moment are main challenges since this research is a student work carried out for the dissertation of master degree with fixed deadline. Though the study attempted for the additional binary classification to evaluate further and to train models without using pre-trained models, it was limited with the extra storage capacity required with this dataset on the server and so was the computational requirement.

CHAPTER 7

CONCLUSION AND FUTURE WORKS

The presence of AI has been vividly seen these days in many real-world applications, because its accuracy has imporved greatly. This is due to many developments, namely the use of deep learning techniques like Convolutional Neural Networks, that have outperformed most other techniques in the field of image classification, recognition, and detection. This thesis presents a review of existing deep CNN models used widely these days for image-classification and recognition with particular focus on the maritime environment. We then used deep CNNs for multi classification of ships and for recognizing their presence in the images generated from videos having varied environmental conditions and captured using different sensors mounted on UAVs at different time periods. The images used came from videos of Seagull dataset. We used four models namely, Xception, VGG16, ResNet50 and InceptionResNetV2 trained on the ImageNet dataset used by the ILSVRC Competition. Instead of training all models from scratch with our data, we used the techniques of transfer learning either freezing all layers (without training with our data) or freezing some layers and training the other with our data. Based on the model´s performance with validation and test data, ResNet 50 has been preferred as the best model although there are only minor differences in the accuracy among all the models. The results of this thesis show that deep learning techniques like CNNs offer good results in maritime applications for ship recognition and classification irrespective of variability on the sea surface and images captured on it by UAVs . So, different environments, platforms, scales or resolutions associated with UAVs can be learned by the CNN models for prediction if the data are properly prepared to train the models with the right parameters. Likewise, transfer learning can help in sharing the learnings of one scenario to another of similar nature and can reduce the time and resources needed to build a completely new model. The integration of deep learning and transfer learning are of great use for developing small

scale models like the output of this thesis. The thesis output generated as the model can not only be used on the past images to have the idea of how ships were used previously but can also be used to monitor ships in the future to learn their pattern and movement behaviors. So, these techniques can be helpful in achieving the essence of maritime monitoring and surveillance for monitoring ships in safe navigation, preventing illegal activities, maintaining border security, safeguarding marine environmental and biological ecosystem from illegal fishing, oil spills, and pollution.
This thesis although limited by time and resource availability has opened the possibility for future research in advancing new applications for the maritime sector in different ways:

- This thesis consists of only classification and recognition of images with respective classes of ships. It does not localize the presence of the ship in an image. Future work can include bounding box detection, or it would be interesting to do mask segmentation of features present in the images as the recent CNN technique called Mask RCNN has been introduced as the outperforming object detection network model.

- Also, the image is trained as if a single ship is present though there are some of the images with two ships of different classes. So, future work should incorporate the presence of multi-label classification.

- Also, this thesis work can be further extended for the real-time image classification aboard UAVs.

- The field of deep learning is changing so rapidly that the used frameworks like tensorflow, Keras, and associated packages are having frequent updates and refinement to offer less complexity in terms of programming algorithms and greater computational efficiency too. So, keeping the work updated with ongoing rapid developments in technology is another challenge to all the research works like this on deep learning discipline.

Bibliographic References

Al-Kaff, A., D. Martín, F. García, A. de la Escalera, and J. M. Armingol (2018). "Survey of computer vision algorithms and applications for unmanned aerial vehicles." In: *Expert Systems with Applications* 92, pp. 447–463.

Ali, A. M. and P. Angelov (2018). "Anomalous behaviour detection based on heterogeneous data and data fusion." In: *Soft Computing* 22.10, pp. 3187–3201. ISSN: 1433-7479. DOI: 10.1007/s00500-017-2989-5. URL: https://doi.org/10.1007/s00500-017-2989-5.

Applications - Keras Documentation. URL: https://keras.io/applications/ (visited on 02/12/2019).

Audebert, N., B. Le Saux, and S. Lefèvre (2017). "Segment-before-detect: Vehicle detection and classification through semantic segmentation of aerial images." In: *Remote Sensing* 9.4, p. 368.

Bejiga, M. B., A. Zeggada, A. Nouffidj, and F. Melgani (2017). "A convolutional neural network approach for assisting avalanche search and rescue operations with uav imagery." In: *Remote Sensing* 9.2, p. 100.

Borghgraef, A., O. Barnich, F. Lapierre, M. Van Droogenbroeck, W. Philips, and M. Acheroy (2010). "An evaluation of pixel-based methods for the detection of floating objects on the sea surface." In: *EURASIP Journal on Advances in Signal Processing* 2010, p. 5.

Borji, A., M.-M. Cheng, H. Jiang, and J. Li (2014). "Salient object detection: A survey. arXiv preprint." In: *arXiv preprint arXiv:1411.5878* 2.4.

Buhl-Mortensen, L. and P. Buhl-Mortensen (2017). "Marine litter in the Nordic Seas: distribution composition and abundance." In: *Marine pollution bulletin* 125.1-2, pp. 260–270.

Chollet, F. et al. (2015). *Keras*. https://keras.io.

Chua, M., D. W. Aha, B. Auslander, K. Gupta, and B. Morris (2014). *Comparison of Object Detection Algorithms on Maritime Vessels*. Tech. rep. NAVAL RESEARCH LAB WASHINGTON DC.

Cruz, G. and A. Bernardino (2016). "Aerial Detection in Maritime Scenarios Using Convolutional Neural Networks." In: *International Conference on Advanced Concepts for Intelligent Vision Systems*. Springer, pp. 373–384.

Deng, L. (2012). "Three classes of deep learning architectures and their applications: a tutorial survey." In: *APSIPA transactions on signal and information processing*.

Deng, Z., H. Sun, S. Zhou, J. Zhao, L. Lei, and H. Zou (2018). "Multi-scale object detection in remote sensing imagery with convolutional neural networks." In: *ISPRS Journal of Photogrammetry and Remote Sensing*.

Ediang, O. and A. Ediang (2013). "Beyond data regulation: finding a solution to the persistent problem of marine debris and sea surface temperature measurement along the coastline of Lagos, Nigeria." In: *Data Science Journal* 12, WDS129–WDS133.

Gallego, A.-J., A. Pertusa, and P. Gil (2018). "Automatic Ship Classification from Optical Aerial Images with Convolutional Neural Networks." In: *Remote Sensing* 10.4, p. 511.

Goodfellow, I., Y. Bengio, A. Courville, and Y. Bengio (2016). *Deep learning*. Vol. 1. MIT press Cambridge.

Griffiths, D. and J. Boehm (2018). "Rapid object detection systems, utilising deep learning and unmanned aerial systems (UAS) for civil engineering applications." In: *International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences-ISPRS Archives*. Vol. 42. 2. International Society for Photogrammetry and Remote Sensing (ISPRS), pp. 391–398.

Guo, J. and C. R. Zhu (2012). "A novel method of ship detection from spaceborne optical image based on spatial pyramid matching." In: *Applied Mechanics and Materials*. Vol. 190. Trans Tech Publ, pp. 1099–1103.

Guo, W. Y., X. F. Wang, and X. Z. Xia (2015). "A remote sensing ship recognition method based on co-training model." In: *Applied Mechanics and Materials*. Vol. 713. Trans Tech Publ, pp. 2077–2080.

Hartemink, M (2012). "Robust automatic object detection in a maritime environment: polynomial background estimation and the reduction of false detections by means of classification." In:

Hong, Z., Q. Jiang, H. Guan, and F. Weng (2007). "Measuring overlap-rate in hierarchical cluster merging for image segmentation and ship detection." In: *Fuzzy Systems and Knowledge Discovery, 2007. FSKD 2007. Fourth International Conference on*. Vol. 4. IEEE, pp. 420–425.

Hu, Y. and Y. Wu (2008). "Number estimation of small-sized ships in remote sensing image based on cumulative projection curve." In: *Audio, Language and Image Processing, 2008. ICALIP 2008. International Conference on*. IEEE, pp. 1522–1526.

Huang, S., H. Xu, and X. Xia (2015). "A remote sensing ship recognition using random forest." In: *4th International Conference on Information Science and Cloud Computing (ISCC)*, p. 11.

Ji-yang, Y., H. Dan, W. Lu-yuan, G. Jian, and W. Yan-hua (2016). "A real-time onboard ship targets detection method for optical remote sensing satellite." In: *2016 IEEE 13th International Conference on Signal Processing (ICSP)*. IEEE, pp. 204–

208. ISBN: 978-1-5090-1344-9. DOI: 10.1109/ICSP.2016.7877824. URL: http://ieeexplore.ieee.org/document/7877824/.

Johnston, D. W. (2019). "Unoccupied Aircraft Systems in Marine Science and Conservation." In: *Annual Review of Marine Science* 11.1, annurev–marine–010318–095323. ISSN: 1941-1405. DOI: 10.1146/annurev-marine-010318-095323. URL: https://www.annualreviews.org/doi/10.1146/annurev-marine-010318-095323.

Kadyrov, A., H. Yu, and H. Liu (2013). "Ship detection and segmentation using image correlation." In: *Systems, Man, and Cybernetics (SMC), 2013 IEEE International Conference on*. IEEE, pp. 3119–3126.

Kanjir, U., H. Greidanus, and K. Oštir (2018). "Vessel detection and classification from spaceborne optical images: A literature survey." In: *Remote Sensing of Environment* 207.December 2017, pp. 1–26. ISSN: 00344257. DOI: 10.1016/j.rse.2017.12.033.

Karpathy, A. "Stanford University CS231n: Convolutional Neural Networks for Visual Recognition." In: (). URL: http://cs231n.stanford.edu/syllabus.html.

Khellal, A., H. Ma, and Q. Fei (2018). "Convolutional Neural Network Based on Extreme Learning Machine for Maritime Ships Recognition in Infrared Images." In: *Sensors* 18.5, p. 1490.

Kim, Y. (2014). "Convolutional neural networks for sentence classification." In: *arXiv preprint arXiv:1408.5882*.

Kornblith, S., J. Shlens, and Q. V. Le (2018). "Do Better ImageNet Models Transfer Better?" In: *arXiv preprint arXiv:1805.08974*.

Krizhevsky, A., I. Sutskever, and G. E. Hinton (2012). "ImageNet Classification with Deep Convolutional Neural Networks." In: *Advances in Neural Information Processing Systems 25*. Ed. by F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger. Curran Associates, Inc., pp. 1097–1105. URL: http://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf.

Kumar, A. S. and E. Sherly (2017). "A convolutional neural network for visual object recognition in marine sector." In: *Convergence in Technology (I2CT), 2017 2nd International Conference for*. IEEE, pp. 304–307.

Lan, J. and L. Wan (2009). "Automatic ship target classification based on aerial images." In: *2008 International Conference on Optical Instruments and Technology: Optical Systems and Optoelectronic Instruments*. Vol. 7156. International Society for Optics and Photonics, p. 715612.

Leclerc, M., R. Tharmarasa, M. C. Florea, A.-C. Boury-Brisset, T. Kirubarajan, and N. Duclos-Hindié (2018). "Ship Classification Using Deep Learning Techniques for Maritime Target Tracking." In: *2018 21st International Conference on Information Fusion (FUSION)*. IEEE, pp. 737–744.

LeCun, Y., L. Bottou, Y. Bengio, and P. Haffner (1998). "Gradient-based learning applied to document recognition." In: *Proceedings of the IEEE* 86.11, pp. 2278–2324.

Leira, F. S., T. A. Johansen, and T. I. Fossen (2015). "Automatic detection, classification and tracking of objects in the ocean surface from uavs using a thermal camera." In: *Aerospace Conference, 2015 IEEE*. IEEE, pp. 1–10.

Li, N., Q. Zhang, H. Zhao, C. Dong, and L. Meng (2016). "Ship detection in high spatial resolution remote sensing image based on improved sea-land segmentation." In: *Hyperspectral Remote Sensing Applications and Environmental Monitoring and Safety Testing Technology*. Vol. 10156. International Society for Optics and Photonics, 101560T.

Li, Q., L. Mou, Q. Liu, Y. Wang, and X. X. Zhu (2018). "HSF-Net: Multiscale deep feature embedding for ship detection in optical remote sensing imagery." In: *IEEE Transactions on Geoscience and Remote Sensing* 99, pp. 1–15.

Li, Y., G. Liu, and S. Chen (2017). "Detection of Moving Object in Dynamic Background Using Gaussian Max-Pooling and Segmentation Constrained RPCA." In: *arXiv preprint arXiv:1709.00657*.

Li-xiaa, X., L. Taoa, and W. Zuo-chengb (2010). "Adaptive Canny edge detection algorithm [J]." In: *Application Research of Computers* 9, pp. 3588–3590.

Lin, Y., H. He, H.-M. Tai, F. Chen, and Z. Yin (2017). "Rotation and scale invariant target detection in optical remote sensing images based on pose-consistency voting." In: *Multimedia Tools and Applications* 76.12, pp. 14461–14483.

Liu, Y., H.-Y. Cui, Z. Kuang, and G.-Q. Li (2017a). "Ship Detection and Classification on Optical Remote Sensing Images Using Deep Learning." In: *ITM Web of Conferences*. Vol. 12. EDP Sciences, p. 05012.

Liu, Z., L. Yuan, L. Weng, and Y. Yang (2017b). "A High Resolution Optical Satellite Image Dataset for Ship Recognition and Some New Baselines." In: *ICPRAM*, pp. 324–331.

McDonnell, M. and A. Lewis (1978). "Ship detection from LANDSAT imagery." In: *Photogrammetric Engineering and Remote Sensing* 44.3.

Miličević, M., K. Zubrinic, I. Obradovic, and T. Sjekavica (2018). "Data Augmentation and Transfer Learning for Limited Dataset Ship Classification." In: *WSEAS Transactions on Systems and Control* 13, pp. 460–465.

Nie, G.-H., P. Zhang, X. Niu, Y. Dou, and F. Xia (2017). "Ship Detection Using Transfer Learned Single Shot Multi Box Detector." In: *ITM Web of Conferences*. Vol. 12. EDP Sciences, p. 01006.

Pan, S. J., Q. Yang, et al. (2010). "A survey on transfer learning." In: *IEEE Transactions on knowledge and data engineering* 22.10, pp. 1345–1359.

Pegler, K., D. Coleman, R. Pelot, and C. P. Keller (2007). "An Enhanced Spatio-spectral Template for Automatic Small Recreational Vessel Detection." In: *Photogrammetric Engineering & Remote Sensing* 73.1, pp. 79–87. ISSN: 00991112. DOI: 10.14358/

PERS . 73 . 1 . 79. URL: http : / / openurl . ingenta . com / content / xref ? genre = article{\&}issn=0099-1112{\&}volume=73{\&}issue=1{\&}spage=79.

Pietkiewicz, T. and J. Matuszewski (2018). "Recognition of maritime objects based on FLIR images using the method of eigenimages." In: *Advanced Trends in Radioelecrtronics, Telecommunications and Computer Engineering (TCSET), 2018 14th International Conference on*. IEEE, pp. 921–929.

Prasad, D. K., C. K. Prasath, D. Rajan, L. Rachmawati, E Rajabaly, and C. Quek (2016). "Challenges in video based object detection in maritime scenario using computer vision." In: *arXiv preprint arXiv:1608.01079*.

Rawat, W. and Z. Wang (2017). "Deep convolutional neural networks for image classification: A comprehensive review." In: *Neural computation* 29.9, pp. 2352–2449.

Razavian, A. S., H. Azizpour, J. Sullivan, and S. Carlsson (2014). "CNN Features off-the-shelf: an Astounding Baseline for Recognition." In: *CoRR* abs/1403.6382. arXiv: 1403.6382. URL: http://arxiv.org/abs/1403.6382.

Ren, S., K. He, R. Girshick, and J. Sun (2017). "Faster R-CNN: towards real-time object detection with region proposal networks." In: *IEEE Transactions on Pattern Analysis & Machine Intelligence* 6, pp. 1137–1149.

Ribeiro, R., G. Cruz, J. Matos, and A. Bernardino (2017). "A Dataset for Airborne Maritime Surveillance Environments." In: *IEEE Transactions on Circuits and Systems for Video Technology*.

Seymour, A., J Dale, M Hammill, P. Halpin, and D. Johnston (2017). "Automated detection and enumeration of marine wildlife using unmanned aircraft systems (UAS) and thermal imagery." In: *Scientific reports* 7, p. 45127.

Shin, H., H. R. Roth, M. Gao, L. Lu, Z. Xu, I. Nogues, J. Yao, D. J. Mollura, and R. M. Summers (2016). "Deep Convolutional Neural Networks for Computer-Aided Detection: CNN Architectures, Dataset Characteristics and Transfer Learning." In: *CoRR* abs/1602.03409. arXiv: 1602.03409. URL: http://arxiv.org/abs/1602.03409.

Simonyan, K. and A. Zisserman (2014). "Very deep convolutional networks for large-scale image recognition." In: *arXiv preprint arXiv:1409.1556*.

Srivastava, N., G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov (2014). "Dropout: a simple way to prevent neural networks from overfitting." In: *The Journal of Machine Learning Research* 15.1, pp. 1929–1958.

Tang, J., C. Deng, G.-B. Huang, and B. Zhao (2015). "Compressed-domain ship detection on spaceborne optical image using deep neural network and extreme learning machine." In: *IEEE Transactions on Geoscience and Remote Sensing* 53.3, pp. 1174–1185.

Turner, J., K. M. Gupta, and D. Aha (2016). "SPARCNN: Spatially related convolutional neural networks." In: *Applied Imagery Pattern Recognition Workshop (AIPR), 2016 IEEE*. IEEE, pp. 1–6.

Valavanidis, A. and T. Vlachogianni (2012). "Marine litter: man-made solid waste pollution in the Mediterranean Sea and coastline. Abundance, composition and sources identification." In: *Environmental Chemistry, Toxicology and Ecotoxicology Resources*.

Ventura, D., A. Bonifazi, M. Gravina, A. Belluscio, and G. Ardizzone (2018). "Mapping and Classification of Ecologically Sensitive Marine Habitats Using Unmanned Aerial Vehicle (UAV) Imagery and Object-Based Image Analysis (OBIA)." In: *Remote Sensing* 10.9, p. 1331.

Wang, R., S. Zhang, L. Pu, J. Yang, C. Yang, J. Chen, C. Guan, Q. Wang, D. Chen, B. Fu, and Others (2016). "Gully erosion mapping and monitoring at multiple scales based on multi-source remote sensing data of the Sancha River Catchment, Northeast China." In: *ISPRS International Journal of Geo-Information* 5.11, p. 200.

Wang, R., J. Li, Y. Duan, H. Cao, and Y. Zhao (2018). "Study on the Combined Application of CFAR and Deep Learning in Ship Detection." In: *Journal of the Indian Society of Remote Sensing* 46.9, pp. 1413–1421.

Xu, C., D. Zhang, Z. Zhang, and Z. Feng (2014). "BgCut: automatic ship detection from UAV images." In: *The Scientific World Journal* 2014.

Xu, J., K. Fu, and X. Sun (2011). "An invariant generalized hough transform based method of inshore ships detection." In: *Image and Data Fusion (ISIDF), 2011 International Symposium on*. IEEE, pp. 1–4.

Yang, X., H. Sun, X. Sun, M. Yan, Z. Guo, and K. Fu (2018). "Position Detection and Direction Prediction for Arbitrary-Oriented Ships via Multiscale Rotation Region Convolutional Neural Network." In: *arXiv preprint arXiv:1806.04828*.

Yao, Y., Z. Jiang, H. Zhang, M. Wang, and G. Meng (2016). "Ship detection in panchromatic images: a new method and its DSP implementation." In: *2nd ISPRS International Conference on Computer Vision in Remote Sensing (CVRS 2015)*. Vol. 9901. International Society for Optics and Photonics, 99010Q.

Yokoya, N. and A. Iwasaki (2015). "Object detection based on sparse representation and Hough voting for optical remote sensing imagery." In: *IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens* 8.5, pp. 2053–2062.

Yosinski, J., J. Clune, Y. Bengio, and H. Lipson (2014). "How transferable are features in deep neural networks?" In: *Advances in neural information processing systems*, pp. 3320–3328.

Yu, J.-G., G.-S. Xia, J. Deng, and J. Tian (2015). "Small object detection in forward-looking infrared images with sea clutter using context-driven Bayesian saliency model." In: *Infrared Physics & Technology* 73, pp. 175–183.

Zhang, B., J. Su, D. Xiong, Y. Lu, H. Duan, and J. Yao (2015). "Shallow convolutional neural network for implicit discourse relation recognition." In: *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pp. 2230–2235.

Zheng, Z., L. Xiao, and B. Zhou (2014). "Generic object detection in maritime environment using self-resemblance." In: *Mechatronics and Control (ICMC), 2014 International Conference on*. IEEE, pp. 469–474.

Zhu, C., H. Zhou, R. Wang, and J. Guo (2010). "A novel hierarchical method of ship detection from spaceborne optical image based on shape and texture features." In: *IEEE Transactions on geoscience and remote sensing* 48.9, pp. 3446–3456.

Zuo, Z. and X. Kuang (2011). "An effective method on ship target detection in remote sensing image of complex background." In: *MIPPR 2011: Automatic Target Recognition and Image Analysis*. Vol. 8003. International Society for Optics and Photonics, p. 800312.

ANNEX I

ANNEX: Modification in Model Architecture

I.1 Modification Strategy of VGG16

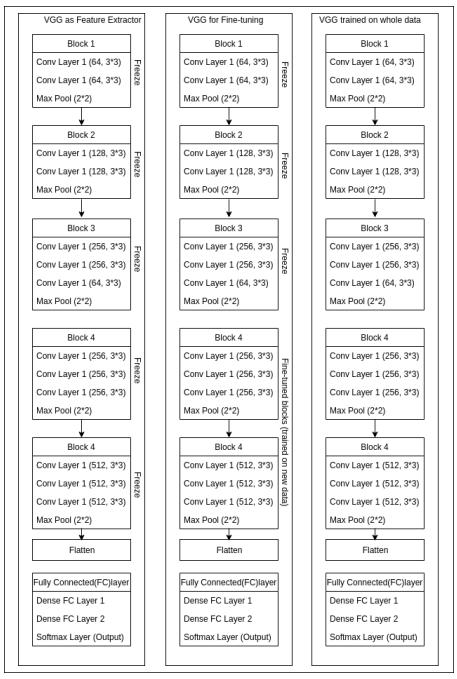


Figure I.1: VGG16 Modification Strategy

ANNEX II

ANNEX 2: Accuracy-Validation Comparison with Hyperparameters

II.1 Training and Validation Accuracy with Xception Model

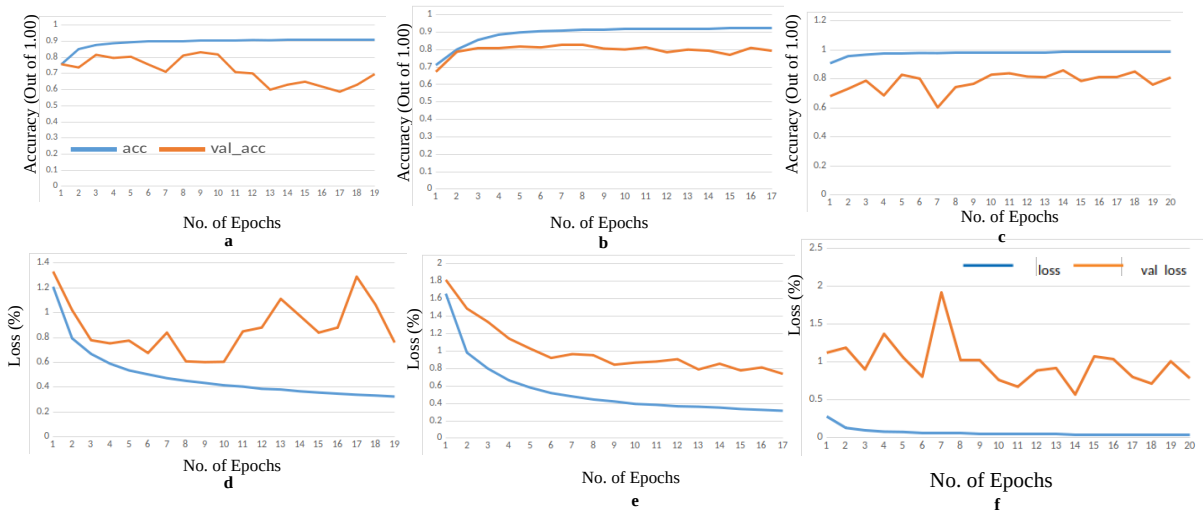


Figure II.1: Accuracy and Loss obtained on Xception model while experimenting with different hyperparameters: (a) with L1 regularization of value 0.01, (b) L2 regularization of value 0.01, (c) with batch normalization layer and (d), (e) and (f) are the corresponding losses.

II.2 Training and Validation Accuracy with InceptionResNetV2 Model

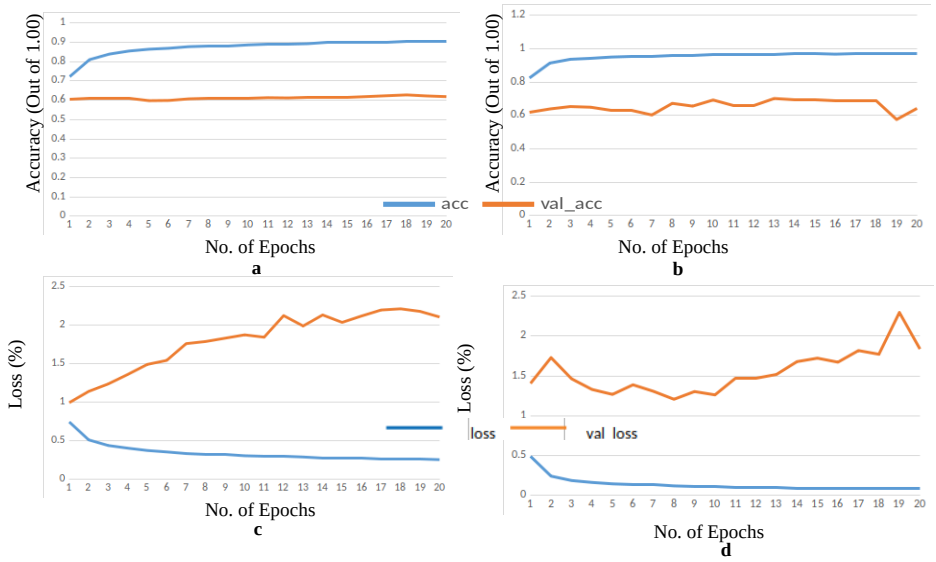


Figure II.2: Accuracy and Loss obtained on InceptionResNetV2 model while experimenting with different hyperparameters: (a) freezing all layers before 777 and learning rate of 2e-04 (b) freezing all layers before 779 and learning rate of 2e-04, (c) represents the loss corresponding to (a) and (d) represents the loss corresponding to (b)

II.3 Training and Validation Accuracy with ResNet50 Model

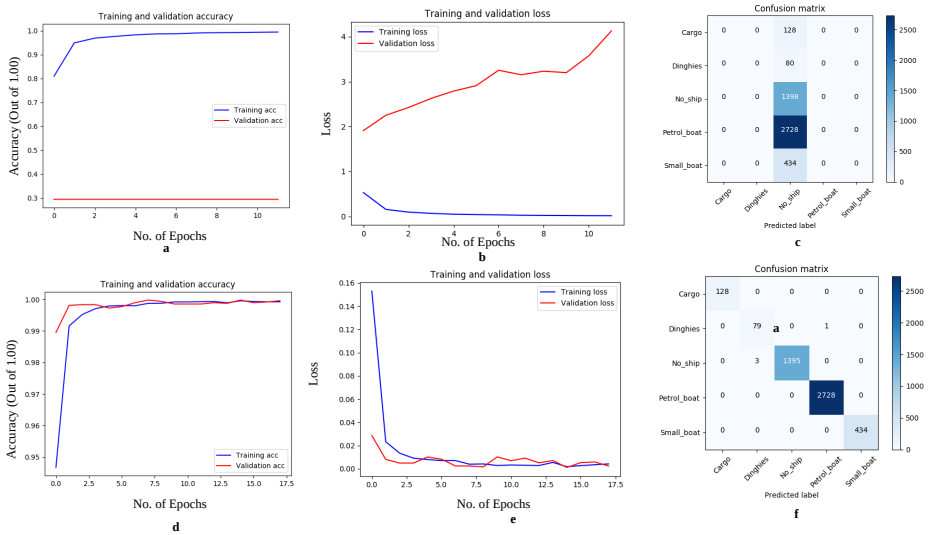


Figure II.3: Accuracy and Loss obtained on ResNetV50 model while experimenting with different hyperparameters: (a) freezing all layers before 163 along with two dense layers of 1024 and 512 followed by dropout layers(b) its corresponding loss, (c) its confusion matrix, (d)freezing all layers and adding a dense layer of 1024 followed by dropout layers before the softmax classifier, (e) its corresponding loss and (f) its confusion matrix

ANNEX III

ANNEX: Algorithm developed to run the model

III.1 Algorithm used for ship classification and recognition

```
  \label{fig:coding}

#This code is prepared to classify and recognize ship s on the
#UAV generated images
#Importing the basic libraries and modules
from __future__ import print_function
import numpy as np
import matplotlib.pyplot as plt
from sklearn.metrics import confusion_matrix
from keras.preprocessing.image import ImageDataGenerator, load_img
from keras.callbacks import ModelCheckpoint, EarlyStopping, CSVLogger
from keras.applications import VGG16
from keras.models import Model
from keras.layers.normalization import BatchNormalization
from keras import models
from keras import layers
from keras.layers import Dense, Flatten
from keras import optimizers
import pandas as pd
import time

start_time = time.time() #Recording the time starts here
#Defining the variables by assigning their path to the directories

train_dir = '/Thesis_Dataset/Train'
validation_dir = '/Thesis_Dataset/validate'
test_dir = '/Thesis_Dataset/Test'
```

```
eval_dir = '/hesis_Dataset/test_evaluation'

image_size = 224

#Load the VGG model, remember to assign false on top layers to remove it and
#avoid the original 1000 classes of ImageNet to modify on our classes (i.e.5)
vgg_conv = VGG16(weights='imagenet', include_top=False, input_shape=
(image_size, image_size, 3))
vgg_conv.summary()

# Freeze/Unfreeze the layers
# Freeze the layers: No specific rule (More as heat and trial)
#but early layers are normally freezed (like edges,)
#whereas latter layers normally extract the specific \\properties of dataset.
#for layer in vgg_conv.layers[:]: # No layers are freezed in this case
#    layer.trainable = True

# Check the trainable status of the individual layers
#for layer in vgg_conv.layers:
#    print(layer, layer.trainable)

# Create the model
model = models.Sequential()
# Add the vgg convolutional base model
model.add(vgg_conv)
# Add new layers
#Dropout value can be changed or removed
#BatchNormalization can be added; it is considered good for robustness
model.add(layers.Flatten())
model.add(layers.Dense(512, activation='relu'))
#model.add(BatchNormalization())
model.add(layers.Dropout(0.5))
#model.add(layers.Dense(512, activation='relu'))
#model.add(layers.Dropout(0.5))
model.add(layers.Dense(5, activation='softmax'))
#Add a layer where input is the output of the  second last layer
#x = vgg_conv.output
#x = Flatten()(x)
#predictions = Dense(5, activation='softmax')(x)
#model = Model(inputs=vgg_conv.input, outputs=predictions)
#model.summary()

#Data augmentation done here, these values can be changed.
train_datagen = ImageDataGenerator(rescale=1./255)
"""
rotation_range=2,
width_shift_range=0.2,
height_shift_range=0.2,
horizontal_flip=True,
fill_mode='nearest')
```

```
"""
test_datagen = ImageDataGenerator(rescale=1./255)

# Change the batchsize according to your system RAM
train_batchsize = 32
val_batchsize = 16
eval_batchsize=1
# Data Generator for Training data
train_generator = train_datagen.flow_from_directory(
train_dir,
target_size=(image_size, image_size),
batch_size=train_batchsize,
class_mode='categorical',
shuffle=True)

# Data Generator for Validation data
validation_generator = test_datagen.flow_from_directory(
validation_dir,
target_size=(image_size, image_size),
batch_size=val_batchsize,
class_mode='categorical',
shuffle=False)
# Compile the model
##Activation Functions like ReLU, Tanh, LeakyReLU are available
model.compile(loss='categorical_crossentropy',
optimizer=optimizers.Adam(lr=1e-4),
metrics=['acc'])
# This is to save the model according to the conditions  and
#stops if validation accuracy is not improving as per the assigned condition
checkpoint = ModelCheckpoint("VGglast.h5", monitor='val_acc',
verbose=1, save_best_only=True, save_weights_only=False, mode='auto',
 period=1)
early = EarlyStopping(monitor='val_acc', min_delta=0, patience=10,
 verbose=1, mode='auto')
csv_logger = CSVLogger("vgglast.csv", append=True)
# Train the Model
history = model.fit_generator(
train_generator,
steps_per_epoch=train_generator.samples//train_generator.batch_size ,
epochs=20 ,
validation_data=validation_generator,
validation_steps=validation_generator.samples//
validation_generator.batch_size,
callbacks = [checkpoint, early, csv_logger])

# Save the Model
#model.save('last_elu4lyr_frze.h5')
import pickle
with open('vgglast', 'wb') as handle: # saving the history of the model
pickle.dump(history.history, handle)
```

```
print("%f seconds" % (time.time() - start_time))
#from keras.utils import plot_model
#plot_model(model, to_file='model.png')
# Plot the accuracy and loss curves
acc = history.history['acc']
val_acc = history.history['val_acc']
loss = history.history['loss']
val_loss = history.history['val_loss']

epochs = range(len(acc))

plt.plot(epochs, acc, 'b', label='Training acc')
plt.plot(epochs, val_acc, 'r', label='Validation acc')
plt.title('Training and validation accuracy')
plt.legend()
plt.savefig('vgglast_acc.png')
plt.figure()

plt.plot(epochs, loss, 'b', label='Training loss')
plt.plot(epochs, val_loss, 'r', label='Validation loss')
plt.title('Training and validation loss')
plt.legend()
plt.savefig('vgglast_loss.png')
plt.show()
#plt.savefig(["plot2.png"])
#Code below is to predict the validation accuracy and
#then to obtain the confusion matrix. Depending upon accuracy on the
#dataset we need we should use the related dataset generator.
# Below is the prediction for validation data, we can switch it for
# test dataset
# as well by copying the code from lines created below for test generator)
#and changing the validation_generator with test_generator

#At first, evaluating the validation data! Should it be test data??
start_time = time.time()
evaluation_generator = test_datagen.flow_from_directory(
eval_dir,
target_size=(image_size, image_size),
batch_size=eval_batchsize,
class_mode='categorical',
shuffle=False)

scores = model.evaluate_generator(evaluation_generator, steps = 596)
print('Loss:', scores[0])
print('Accuracy:', scores[1])

print("%f seconds" % (time.time() - start_time))

start_time = time.time()

```

```
#For predicting accuracy
# Get the filenames from the generator
fnames = evaluation_generator.filenames

# Get the ground truth from generator
ground_truth = evaluation_generator.classes

# Get the label to class mapping from the generator
label2index = evaluation_generator.class_indices

# Getting the mapping from class index to class label
idx2label = dict((v,k) for k,v in label2index.items())

# Get the predictions from the model using the generator
predictions = model.predict_generator(evaluation_generator,
steps=evaluation_generator.samples/
evaluation_generator.batch_size,verbose=1)
pred_class = np.argmax(predictions,axis=1)

errors = np.where(pred_class != ground_truth)[0]
print("No␣of␣errors␣=␣{}/{}".format(len(errors),eval_generator.samples))
"""
#␣Show␣the␣errors
for␣i␣in␣range(len(errors)):
pred_class␣=␣np.argmax(predictions[errors[i]])
pred_label␣=␣idx2label[pred_class]

title␣=␣'Original␣label:{},␣Prediction␣:{},␣confidence␣:␣{:.3f}'.format(
fnames[errors[i]].split('/')[0],
pred_label,
predictions[errors[i]][pred_class])

original␣=␣load_img('{}/{}'.format(validation_dir,fnames[errors[i]]))
plt.figure(figsize=[7,7])
plt.axis('off')
plt.title(title)
plt.imshow(original)
plt.show()
"""
print("%f␣seconds" % (time.time() - start_time))

#To save the prediction in excel sheet but since I am directly predicting
#here Confusion matrix, it was not saved, may be useful in server to save it.
import sklearn.metrics as metrics
#pred_class = np.argmax(prob, axis=1)
target_names = ['cargo', 'no_ship', 'ship_dinghies',
 'small_boat', 'small_boat_patrol']
report = metrics.classification_report(ground_truth,
pred_class, target_names=target_names)
print(report)
```

```
228
229 import itertools
230 #cm = confusion_matrix(idx2label, predictions )
231 def plot_confusion_matrix(cm, classes,
232 normalize=False,
233 title='Confusion␣matrix',
234 cmap=plt.cm.Blues):
235 if normalize:
236 cm = cm.astype('float') / cm.sum(axis=1)[:, np.newaxis]
237 print("Normalized␣confusion␣matrix")
238 else:
239 print('Confusion␣matrix,␣without␣normalization')
240
241 #print(cm)
242
243 plt.imshow(cm, interpolation='nearest', cmap=cmap)
244 plt.title(title)
245 plt.colorbar()
246 tick_marks = np.arange(len(classes))
247 plt.xticks(tick_marks, classes, rotation=45)
248 plt.yticks(tick_marks, classes)
249
250 fmt = '.2f' if normalize else 'd'
251 thresh = cm.max() / 5.
252 for i, j in itertools.product(range(cm.shape[0]), range(cm.shape[1])):
253 plt.text(j, i, format(cm[i, j], fmt),
254 horizontalalignment="center",
255 color="white" if cm[i, j] > thresh else "black")
256
257 plt.ylabel('True␣label')
258 plt.xlabel('Predicted␣label')
259 plt.tight_layout()
260
261
262 # Compute confusion matrix
263 cnf_matrix = confusion_matrix(ground_truth, pred_class)
264 np.set_printoptions(precision=2)
265 # Plot normalized confusion matrix
266 plt.figure()
267 plot_confusion_matrix(cnf_matrix, classes=target_names, normalize=False,
268 title='Confusion␣matrix')
269
270 plt.show()
271 plt.savefig('vgglast.png')
272
273 print("%f␣seconds" % (time.time() - start_time))
274
275
276
277 start_time = time.time()
```

```
278 #To evaluate accuracy of the model with the test data
279 test_generator = test_datagen.flow_from_directory(
280 test_dir,
281 target_size=(image_size, image_size),
282 batch_size=1,
283 class_mode=None,
284 shuffle=False)
285
286
287 test_generator.reset()
288 #Predicting for the test data
289
290 pred=model.predict_generator(test_generator, steps=
291 test_generator.samples//test_generator.batch_size )
292
293 predicted_class_indices=np.argmax(pred, axis=1)
294 labels = (train_generator.class_indices)
295 labels=dict((v,k) for k,v in labels.items())
296 predictions = [labels[k] for k in predicted_class_indices]
297
298 filenames=test_generator.filenames
299 results=pd.DataFrame({"Filename":filenames,
300 "Predictions":predictions})
301 results.to_csv("VGglast.csv", index=True)
302
303 print("%f␣seconds" % (time.time() - start_time))
```

Masters Program in Geospatial Technologies

SHIP RECOGNITION ON THE SEA SURFACE USING AERIAL IMAGES TAKEN BY UAV:

A Deep Learning Approach

Laxmi Thapa

Dissertation submitted in partial fulfilment of the requirements
for the Degree of *Master of Science in Geospatial Technologies*

NOVA IMS Information Management School
Universitat Jaume I
Westfälische Wilhelms-Universität Münster

Masters Program in **Geospatial Technologies**

Supported by:
Education and Culture
ERASMUS MUNDUS